

LangLoc: Fine-Grained Localization using Language Grounding

Master's Project

**Shaurya Kishore Panwar, Roham Zendeheel
Nobari, Shirley Lau Feng Yi, Abu Bakr
Rahman Shaik**

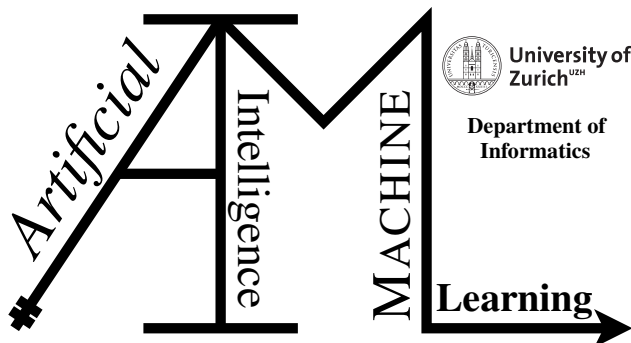
24-744-856, 23-755-432, 23-756-703, 23-756-737

Submitted on
April xx, 2026

Thesis Supervisor

Prof. Dr. Manuel Günther, Prof. Dr. Marc Pollefeys

Dr. Dániel Béla Baráth



ETH zürich



Computer Vision And
Geometry Lab

Declaration of Independence for Written Work

I hereby declare that I have **composed** this work independently and without the use of any aids other than those declared (including generative AI such as ChatGPT) – the use of generative AI to **improve** my composed work was permitted by the thesis supervisor. I am aware that I take full responsibility for the scientific character of the submitted text myself, even if AI aids were used. All passages taken verbatim or in sense from published or unpublished writings are identified as such. The work has not yet been submitted in the same or similar form or in excerpts as part of another examination.

Place, Date

Shaurya Kishore Panwar, Roham Zendehtdel
Nobari, Shirley Lau Feng Yi, Abu Bakr Rahman
Shaik

Master's Project

Author: Shaurya Kishore Panwar, Roham Zendehtdel Nobari, Shirley Lau Feng Yi, Abu Bakr Rahman Shaik

Project period: 06.08.2025 - 15.04.2026

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

Acknowledgements

We are deeply grateful to our supervisors at ETH Zurich, Prof. Dr. Marc Pollefeys and Dr. Dániel Béla Baráth, for their guidance and mentorship throughout this master's project. We want to extend a special and personal thank you to Dr. Baráth. His constant support, patience, and encouragement at every stage were a true anchor for us. This thesis would not have been possible without him.

We also extend our most sincere appreciation to Prof. Dr. Manuel Günther for providing us with the opportunity to work on this highly relevant topic and for providing the resources, environment, and freedom that allowed our research to thrive. His vision and continuous encouragement were instrumental in guiding the direction and ultimate quality of this project.

Abstract

We study language-based indoor localization in known 3D environments. Given a free-form description of what an observer sees, the goal is to solve two coupled tasks: first, retrieve the correct scene from a database of indoor scans, and second, estimate the observer’s intra-scene pose, namely a 2D floor position and viewing direction, from language alone. Each 3D environment is represented as a scene graph, a directed attributed graph in which nodes correspond to semantically labelled object instances, each carrying a 518-dimensional feature vector that concatenates the instance’s metric centroid, mean colour, and a text embedding of its class label, while directed edges encode typed spatial relations (e.g. *left_of*, *above*, *near*) derived from K-nearest-neighbour geometry among object centroids. At query time, a free-form description is parsed into a text scene graph with the same feature schema but zero-padded geometric fields, enabling a shared encoder to operate over both modalities. Scene retrieval is performed by a dual-branch GATv2 encoder that processes geometric edge attributes and embedded relationships, fusing them via a learned per-node sigmoid gate. Trained with a contrastive InfoNCE objective, this improves Top-1 recall by 8 percentage-point improvement over prior work. Fine localization scores a dense floor grid of candidate observer positions by casting rays from each candidate at eye height to every matched object’s centroid on the scene mesh, then searching for the horizontal field-of-view window that explains the largest subset of visible objects, producing a softmax posterior over pose-direction hypotheses with a median position error of 0.95 m. When the resulting posterior is multi-modal, a Bayesian dialog module selects questions that maximize expected information gain over the pose distribution and iteratively refines it via yes/no answers about object visibility and spatial relations. We introduce a new benchmark comprising over 13 000 pose-indexed natural-language descriptions across more than 1 300 indoor 3D scans. Code and data will be released.

Contents

1	Introduction	1
1.1	Motivation and Project Goals	1
1.2	Description of Work	2
1.3	Report Outline	3
2	Theoretical Background	5
2.1	Graph Neural Networks	5
2.1.1	Graph Attention Networks and GATv2	5
2.2	Scene Graph Representations	8
2.3	CLIP: Contrastive Language–Image Pre-training	9
2.4	Contrastive Learning and InfoNCE	10
2.5	Determinantal Point Processes	10
2.6	Geometric Foundations	11
2.6.1	Triangle Meshes	11
2.6.2	Ray Casting	12
2.6.3	Visibility Determination	12
2.6.4	Field of View and Camera Geometry	12
2.7	Bayesian Inference for Sequential Disambiguation	12
3	Related Work	15
3.1	Visual Localization	15
3.2	Language-based and Cross-modal Localization	16
3.3	Dialog-based Localization and Embodied Navigation	17
3.4	3D Scene Graphs and Vision-Language Grounding	17
3.4.1	Text Scene Graphs	18
4	Data	19
4.1	Benchmark Datasets	19
4.2	The LangLoc Dataset	19
4.2.1	Image Quality Assessment Filtering	20
4.2.2	Visibility Computation via GPU Rasterisation	20
4.2.3	Two-Stage DPP frame Selection	20
4.2.4	Description Generation	22
4.2.5	Scene Graph Construction for ScanNet	23

5	Approach	25
5.1	Problem Formulation	25
5.2	Scene Representation	26
5.2.1	3D Scene Graph Construction	26
5.2.2	Graph Preprocessing	27
5.3	Text Representation	29
5.3.1	Text-to-Graph Conversion	29
5.3.2	Feature Encoding	29
5.4	Cross-modal Retrieval Model	30
5.4.1	Graph Encoder Architecture	30
5.4.2	Graph-level Embedding	32
5.4.3	Similarity Function	33
5.4.4	Training Objective	33
5.5	Fine Localization	34
5.5.1	Text-to-Scene Object Matching	34
5.5.2	Candidate Floor Positions and Inside-Scene Filtering	35
5.5.3	Ray-Cast Visibility Scoring	36
5.5.4	Visibility Reasoning and FoV-Consistent Direction Estimation	36
5.5.5	Pose Hypothesis Scoring	37
5.5.6	Density-Based Aggregation of the Final Pose	38
5.6	Dialog-based Disambiguation	39
5.6.1	Reference Frames and Semantic Pooling	39
5.6.2	Frame-Level Posterior (Backend A3)	40
5.6.3	Likelihood Model	41
5.6.4	Bayesian Update	42
5.6.5	Question Selection	42
5.6.6	Stopping Criteria and Final Pose	43
6	Results	45
6.1	Coarse Localization	45
6.1.1	Evaluation protocol	45
6.2	Fine Localization	48
6.2.1	Evaluation Protocol	49
6.2.2	Results on 100-Scene Subsets	49
6.2.3	Full-Scale Evaluation	49
6.2.4	Qualitative Results	50
6.3	Dialog-based Disambiguation	51
7	Discussion	53
7.1	Scene Retrieval	53
7.1.1	Comparison with Text2SGM	53
7.1.2	Role of Unique Instance Labelling	54
7.1.3	Role of Subgraph Augmentation	54
7.1.4	Limitations of Scene Retrieval	54
7.2	Fine Localization	55
7.2.1	The Midpoint Baseline Artefact	55
7.2.2	Visibility Scoring as Implicit Grounding	55
7.2.3	Cross-Dataset Generalisation	55
7.3	Dialog-based Disambiguation	55
7.3.1	Impact of Dialog	55
7.3.2	Limitations and Future Work	56

Contents	vii
8 Conclusion	57
8.1 Summary of Contributions	57
9 Author Contributions	59
A Hyperparameters and Architecture Details	61
A.1 GATv2 Encoder Architecture	61
A.2 Scene Retrieval Training	61
A.3 Retrieval Scoring Weights	62
A.4 Fine-Localization Hyperparameters	62
A.5 Dialog Hyperparameters	62
A.5.1 Likelihood Model	62
A.5.2 Stopping Criteria	63
A.5.3 Question Selection Parameters	64
A.6 Qwen-based Answering Protocol	64
B Dialog Backend Architectures	65
B.1 Backend A1: Candidate Posterior	65
B.2 Backend A2: Particle Filter	65
B.3 Backend A3: Frame Posterior (Default)	66
B.4 Backend Comparison Results	66
List of Figures	69
List of Tables	70
Abbreviations	71

Introduction

Language provides a natural and intuitive way for humans to describe their surroundings. In this work, we investigate the problem of localising an observer within a known 3D environment using only a free-form textual description. The underlying 3D scene database is constructed offline from RGB-D scans with semantic instance segmentation, a one-time process that can be conducted under controlled conditions. However, unlike traditional visual localisation approaches that require the user to capture and transmit an image at every query, our method requires only a short text description at inference time, making the query itself lightweight, privacy-preserving, and independent of any camera or specialised on-device hardware.

We propose *LangLoc*, a pipeline that bridges natural language and structured 3D scene representations to enable both scene-level retrieval and fine-grained intra-scene localisation, including an optional dialog-based refinement step for ambiguous descriptions.

1.1 Motivation and Project Goals

Knowing where you are is fundamental to almost every location-aware service: indoor navigation, robot assistance, augmented reality, and emergency response all require an accurate pose estimate. The dominant localisation paradigm today is visual: a device captures an image or video stream, uploads it to a server, and receives a pose estimate in return (Sattler et al., 2017; Sarlin et al., 2019). While effective, this approach carries significant drawbacks at query time. Each localisation request requires the user to capture and transmit an image of their current surroundings, which is bandwidth-heavy and may inadvertently expose sensitive information — bystanders, personal belongings, or confidential documents visible in the frame. Our method shares the requirement of an offline 3D capture for map construction, a one-time process that can be conducted under controlled conditions. However, at query time, the user transmits only a short text description, which is orders of magnitude smaller than an image and inherently avoids capturing visual information about the environment’s current state.

Language offers a compelling alternative. Describing a scene in natural language, e.g. “I’m standing in front of a bookshelf, with a blue sofa on my left and a TV across the room,” is natural, fast, and transmits minimal personally identifiable information. A text description is orders of magnitude smaller than an image, requires no camera or special hardware, and mirrors how people communicate spatial information in everyday life.

Despite this appeal, language-based localisation remains largely unsolved. Existing approaches primarily focus on coarse scene retrieval, identifying which room in a database a description refers to. (Kolmet et al., 2022; Chen et al., 2024) However, resolving a precise pose within a scene from language alone is significantly more challenging. Many viewpoints within the same room



Figure 1.1: LANGLOC OVERVIEW. Input: a free-form description mentioning objects such as fireplace, couch, pillow, and fan. Scene retrieval (Section 5.4) ranks candidate 3D scenes; the correct scene (green, ✓) is identified from a pool of candidates. Fine localisation (Section 5.5) scores a dense floor grid by ray-cast object visibility, producing a heat map over candidate positions; the predicted pose (green frustum) is compared against the ground truth (red frustum). Dialog disambiguation (Section 5.6) asks targeted yes/no questions (as illustrated in the figure) and iteratively refines the posterior until it concentrates on the correct pose.

share similar semantic descriptions, differing only in subtle geometric or visibility cues that are difficult to capture in text.

The goal of this project is therefore to investigate how natural language can be used not only to retrieve a scene but also to localise an observer within it. In particular, we aim to bridge the gap between high-level semantic descriptions and fine-grained spatial reasoning.

1.2 Description of Work

We present *LangLoc*, the first pipeline for fine-grained indoor localisation from natural language. The system operates on a database of 3D indoor scenes, each represented as a scene graph, a directed attributed graph whose nodes correspond to semantically labelled object instances and whose edges encode typed spatial relations derived from the scene geometry. At query time, a free-form natural language description is parsed into a text scene graph with the same structural schema, enabling cross-modal matching through a shared encoder.

The pipeline proceeds in three stages. In the first stage, scene retrieval, the text scene graph is matched against the database of 3D scene graphs using a dual-branch GATv2 encoder. One branch processes geometric information, relative displacements and metric extents between object instances, while the other processes semantic relational information via CLIP-embedded relation strings. The two branches are fused through a learned per-node sigmoid gate, and the encoder is trained with an InfoNCE contrastive objective on pairs of 3D scene graphs, generalis-

ing to text graph queries at inference through a shared feature space with zero-padded geometric fields.

In the second stage, fine localisation, the system estimates the observer’s 2D floor position and viewing direction within the retrieved scene. A dense grid of candidate positions is sampled on the floor plane at a fixed eye height, and for each candidate, rays are cast to the centroids of matched objects on the scene mesh to determine visibility. A field-of-view sweep then identifies, for each candidate, the viewing direction that explains the largest subset of simultaneously visible objects. Candidates are scored by combining visibility support with a proximity term and aggregated into a softmax posterior over pose-direction hypotheses.

In the third stage, dialog-based disambiguation, the system addresses cases where the posterior remains multi-modal – that is, where the description is consistent with multiple distinct locations. A Bayesian dialog module selects yes/no questions about object visibility or spatial relations that maximise expected information gain over the current pose distribution, iteratively updating the posterior with each answer until the estimate converges.

This three-stage design raises several research questions that we investigate in this work. First, can structured scene graph matching from language alone achieve retrieval accuracy competitive with or superior to existing cross-modal methods? Second, to what extent can ray-cast visibility reasoning on a floor grid resolve a precise intra-scene pose from a textual description, given that many viewpoints within the same room share similar semantic content? Third, does interactive dialog with information-theoretic question selection measurably reduce localisation ambiguity compared to single-shot estimation?

1.3 Report Outline

Chapter 2 introduces the theoretical foundations underlying our approach, covering graph neural networks, the GATv2 attention mechanism, CLIP embeddings, contrastive learning, ray-casting, and Bayesian inference. Chapter 3 surveys related work in visual localisation, language-based localisation, dialog-based navigation, and 3D scene graphs. Chapter 5 presents our approach in detail, including scene graph construction, the dual-branch retrieval encoder, visibility-based fine localisation, and dialog-based disambiguation. Chapter 6 reports quantitative and qualitative results for all three pipeline stages. Chapter 7 discusses the findings, analyses key design decisions, and identifies limitations. Finally, Chapter 8 summarises the work and outlines directions for future research.

Theoretical Background

This chapter introduces the technical foundations underlying our approach. We cover graph neural networks and the GATv2 attention mechanism used in our scene retrieval encoder, the CLIP vision-language model that provides semantic embeddings, the contrastive learning framework used for training, the geometric principles behind ray-cast visibility scoring, and the Bayesian inference formulation that drives our dialog-based disambiguation module.

2.1 Graph Neural Networks

Graph Neural Networks (GNNs) extend deep learning to data with irregular, non-Euclidean structure [Zhou et al. \(2020\)](#). A graph $G = (V, E)$ consists of a set of nodes V and edges E ; each node v_i carries an initial feature vector $\mathbf{h}_i^{(0)} \in \mathbb{R}^{d_0}$, where the subscript denotes layer 0, since the feature dimensionality may change across layers as each layer l projects to $\mathbb{R}^{d_l}$. GNNs update these features through iterative *message passing* [Gilmer et al. \(2017\)](#): at each layer l , every node aggregates information from its neighbourhood and transforms the result to produce an updated representation.

The general message-passing framework can be written as:

$$\mathbf{m}_i^{(l)} = \bigoplus_{j \in \mathcal{N}(i)} \psi^{(l)}(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \boldsymbol{\xi}_{ij}), \quad (2.1)$$

$$\mathbf{h}_i^{(l+1)} = \gamma^{(l)}(\mathbf{h}_i^{(l)}, \mathbf{m}_i^{(l)}), \quad (2.2)$$

where $\mathcal{N}(i)$ denotes the set of neighbours of node i , $\psi^{(l)}$ is a learnable message function, $\bigoplus$ is a permutation-invariant aggregation operator (e.g. sum, mean, or max), $\boldsymbol{\xi}_{ij}$ is an optional edge feature, and $\gamma^{(l)}$ is a learnable update function. After L layers, each node embedding $\mathbf{h}_i^{(L)}$ encodes information from its L -hop neighbourhood.

Different GNN variants arise from specific choices of ψ , $\bigoplus$, and γ . Graph Convolutional Networks (GCNs) [Kipf and Welling \(2017\)](#) use a normalised sum aggregation with a shared linear transformation.

2.1.1 Graph Attention Networks and GATv2

Graph Attention Networks (GATs) [Veličković et al. \(2018\)](#) replace the fixed normalised weighting of GCNs with learned attention coefficients. For a node i and its neighbour j at layer l , the original

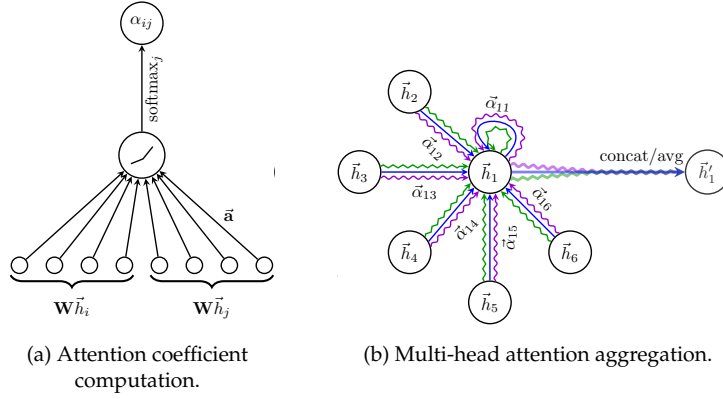


Figure 2.1: GRAPH ATTENTION MECHANISM. In subfigure (a), the attention coefficient α_{ij} is computed by applying a shared attention vector $\mathbf{a}$ to the concatenation of linearly projected features $\mathbf{W}\mathbf{h}_i$ and $\mathbf{W}\mathbf{h}_j$, followed by LeakyReLU and softmax normalisation. Subfigure (b) illustrates multi-head attention with $K = 3$ heads on node 1's neighbourhood: different arrow styles and colours denote independent attention computations, and the aggregated features from each head are concatenated or averaged to produce $\mathbf{h}'_1$. Figure from Veličković et al. (2018).

GAT computes an unnormalised attention score as:

$$e_{ij}^{(l)} = \text{LeakyReLU}(\mathbf{a}^{(l)\top} [\mathbf{W}^{(l)}\mathbf{h}_i^{(l)} \parallel \mathbf{W}^{(l)}\mathbf{h}_j^{(l)}]), \quad (2.3)$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d' \times d_l}$ is a shared linear projection, $\mathbf{a}^{(l)} \in \mathbb{R}^{2d'}$ is a learnable attention weight vector that scores the relevance of neighbour j to node i , and $[\cdot \parallel \cdot]$ denotes concatenation. The scores are normalised across the neighbourhood using softmax:

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})}. \quad (2.4)$$

The updated node representation is then:

$$\mathbf{h}_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} \mathbf{W}^{(l)}\mathbf{h}_j^{(l)} \right), \quad (2.5)$$

where σ is a nonlinearity. Multi-head attention runs H independent attention heads in parallel and concatenates (or averages) their outputs. Figure 2.1 illustrates the attention mechanism and multi-head aggregation.

The Static Attention Problem

Brody et al. (2022) identified a fundamental limitation of the original GAT: because the attention vector $\mathbf{a}$ is applied *after* the concatenation of the linearly projected features $\mathbf{W}\mathbf{h}_i$ and $\mathbf{W}\mathbf{h}_j$, the attention score in (2.3) can be decomposed as:

$$e_{ij} = \text{LeakyReLU}(\mathbf{a}_1^\top \mathbf{W}\mathbf{h}_i + \mathbf{a}_2^\top \mathbf{W}\mathbf{h}_j), \quad (2.6)$$

where $\mathbf{a}_1$ and $\mathbf{a}_2$ are the first and second halves of $\mathbf{a}$. The key insight is that the monotonicity of LeakyReLU means the ranking of attention scores e_{ij} over neighbours j is determined entirely by

$\mathbf{a}_2^\top \mathbf{W} \mathbf{h}_j$, independent of the query node i (the term $\mathbf{a}_1^\top \mathbf{W} \mathbf{h}_i$ is constant across all j). This means GAT computes a *static* attention ranking: the relative importance of neighbours does not change depending on which node is querying them.

GATv2: Dynamic Attention

GATv2 [Brody et al. \(2022\)](#) resolves this limitation through two changes: it replaces the single shared projection $\mathbf{W}$ with separate query and key projections $\mathbf{W}_l$ and $\mathbf{W}_r$ whose outputs are *summed* rather than concatenated, and it applies the nonlinearity *before* the dot product with the attention vector. Together, these make the attention function strictly more expressive:

$$e_{ij} = \mathbf{a}^\top \text{LeakyReLU}(\mathbf{W}_l \mathbf{h}_i + \mathbf{W}_r \mathbf{h}_j), \quad (2.7)$$

where $\mathbf{W}_l, \mathbf{W}_r \in \mathbb{R}^{d' \times d}$ are separate projections for the query and key nodes, and $\mathbf{a} \in \mathbb{R}^{d'}$. Because the LeakyReLU is applied to the *sum* of the two projections, the nonlinearity mixes the contributions of $\mathbf{h}_i$ and $\mathbf{h}_j$ before the dot product with $\mathbf{a}$, enabling *dynamic* attention: the ranking of neighbours depends on the identity of the query node.

When edge features $\xi_{ij} \in \mathbb{R}^{d_e}$ are available, the GATv2 attention mechanism can be extended to incorporate them by projecting the edge feature and adding it inside the nonlinearity. This edge-conditioned variant computes:

$$e_{ij} = \mathbf{a}^\top \text{LeakyReLU}(\mathbf{W}_l \mathbf{h}_i + \mathbf{W}_r \mathbf{h}_j + \mathbf{W}_e \xi_{ij}), \quad (2.8)$$

where $\mathbf{W}_e \in \mathbb{R}^{d' \times d_e}$ projects the edge feature to the attention dimension. This edge-conditioned form is essential for our pipeline, where the geometric branch conditions on spatial displacement vectors and the relation branch conditions on CLIP-embedded relation strings.

The softmax normalisation and multi-head extension follow the same structure as the original GAT (2.4)–(2.5). GATv2 has been shown to achieve strictly higher expressiveness than GAT on tasks where the optimal attention pattern depends on the query [Brody et al. \(2022\)](#). Figure 2.2 illustrates the difference empirically.

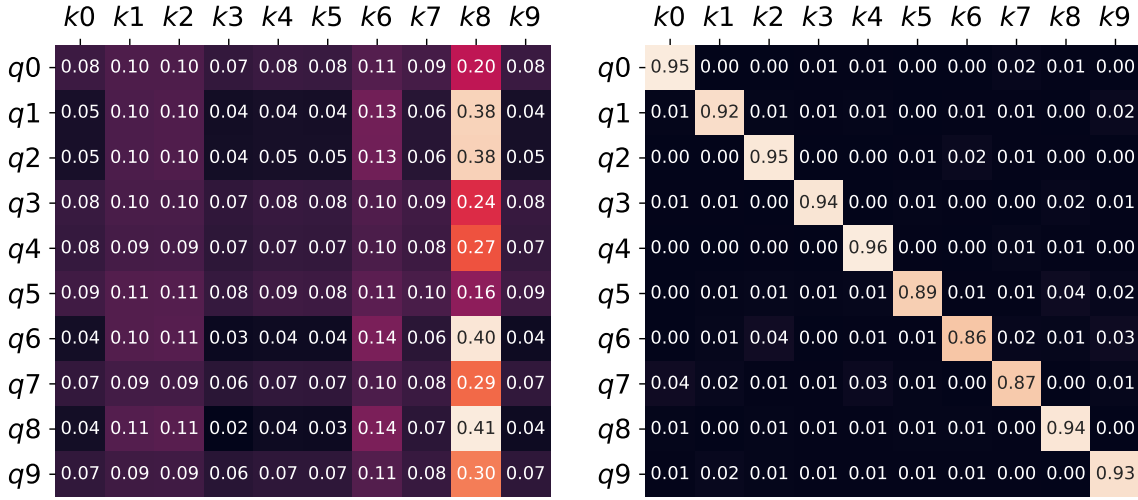


Figure 2.2: STATIC VS. DYNAMIC ATTENTION. Attention coefficients on a complete bipartite graph with query nodes $\{q_0, \dots, q_9\}$ and key nodes $\{k_0, \dots, k_9\}$. Left (GAT): All queries attend mostly to key k_8 ; the attention ranking is static and independent of the query. Right (GATv2): Each query attends to a different key along the diagonal; the attention ranking is dynamic and query-dependent. Figure from [Brody et al. \(2022\)](#).

2.2 Scene Graph Representations

A 3D scene graph is a graph-structured summary of a static 3D scan, in which nodes correspond to object instances and edges encode pairwise spatial relations between them. The formulation adopted throughout this thesis is the one introduced by the 3D Semantic Scene Graphs (3DSSG) dataset [Wald et al. \(2020\)](#), which provides scene graphs for a large number of reconstructed indoor rooms together with a fixed label and relation vocabulary.

3D scene graph. Formally, a 3D scene graph is a directed attributed graph

$$G^s = (V^s, E^s), \quad (2.9)$$

associated with a single reconstructed scene. Each node $v_i \in V^s$ represents one object instance identified by the scan’s segmentation, and carries the following attributes:

- a unique integer instance identifier,
- a semantic class label drawn from a fixed label vocabulary (e.g. *chair*, *window*, *fridge*),
- an oriented bounding box consisting of a centroid $\mu_i \in \mathbb{R}^3$, three orthonormal axes, and per-axis half-extents,
- optional descriptive attributes grouped into a small set of categories such as *colour*, *material*, *shape*, and *state*; for example a node labelled *chair* may carry *colour*:red and *material*:wooden.

Each directed edge $(v_i, v_j) \in E^s$ is labelled with a relation drawn from a fixed vocabulary of typed predicates. 3DSSG uses a closed set of roughly forty relations grouped into directional (*left*, *right*, *front*, *behind*, *higher than*, *lower than*), support (*standing on*, *supported by*), proximity (*close by*), and comparative (*bigger than*, *smaller than*, *same colour as*, ...) families. Edges are directed: the triple

$(v_i, v_j, \text{left of})$ asserts that v_i is to the left of v_j , and the inverse $(v_j, v_i, \text{right of})$ is stored as a separate edge. A scene graph therefore encodes both *what* objects a room contains (nodes) and *how they are arranged* relative to one another (edges).

Text scene graph. A *text scene graph* [Chen et al. \(2024\)](#) is the same kind of object extracted from a natural-language description rather than from 3D geometry. Given a query sentence T , a parser produces a directed attributed graph

$$G^t = (V^t, E^t), \quad (2.10)$$

whose nodes $v_i \in V^t$ represent the objects mentioned in T and whose edges E^t represent the relational statements appearing in T . Nodes carry a semantic label and any attributes explicitly asserted by the text (*a small round table* yields the attributes `size:small`, `shape:round`), but have no geometric fields such as centroid or bounding box, because the text alone does not pin down 3D coordinates. Edges carry a relation name drawn from the same predicate vocabulary as the 3D scene graph when such a relation is available, and are otherwise kept as free-form natural-language labels. Scene graphs and text scene graphs are therefore two instances of the same abstract data structure, differing only in which node and edge attributes are populated and in how those attributes are obtained.

2.3 CLIP: Contrastive Language–Image Pre-training

CLIP [Radford et al. \(2021b\)](#) is a vision–language foundation model that learns a shared embedding space for images and text via contrastive pre-training on 400 million image–text pairs collected from the internet. It consists of two parallel encoders: an image encoder (a Vision Transformer or ResNet variant) and a text encoder (a Transformer), both mapping their respective inputs to a shared d -dimensional space ($d = 512$ for ViT-B/32). A key property of the CLIP text encoder, compared with text-only encoders such as BERT, is that its representations are grounded in visual semantics through the joint training objective. This means that words describing spatial and visual concepts (e.g. object categories, spatial prepositions) are embedded in a space that reflects their visual co-occurrence patterns, not just their distributional statistics in text corpora.

Given a batch of N image–text pairs, CLIP maximises the cosine similarity between matching pairs while minimising it for all $N^2 - N$ non-matching pairs (Figure 2.3). The result is an embedding space where semantically related concepts, whether expressed as images or text, are nearby. In our pipeline, we use only the text encoder $\phi(\cdot)$, which maps an input string to a unit-normalised vector $\phi(s) \in \mathbb{R}^{512}$.

The resulting embedding space has the property that semantically related concepts, whether expressed as images or text, are mapped to nearby points. This enables zero-shot transfer: a text string and an image depicting the same concept will have high cosine similarity even if that specific pairing was never seen during training. The text encoder $\phi(\cdot)$ maps an input string to a unit-normalised vector $\phi(s) \in \mathbb{R}^{512}$, and semantic similarity between different phrasings of the same concept is naturally preserved in this space.

The ability of CLIP to capture semantic similarity across different phrasings is critical for cross-modal matching, where training-time relation predicates (e.g. `left_of`) may differ from test-time predicates in natural language descriptions (e.g. `to the left of`).

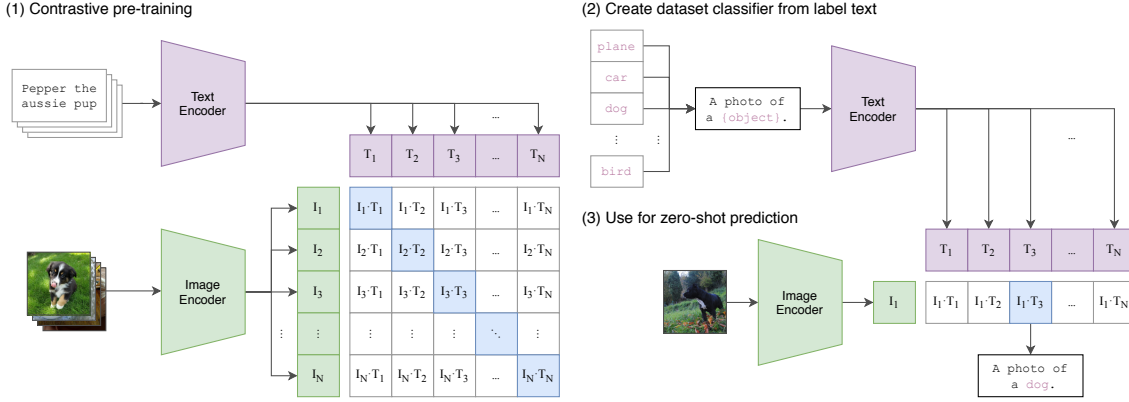


Figure 2.3: CLIP CONTRASTIVE PRE-TRAINING. An image encoder and a text encoder are jointly trained to maximise cosine similarity for matching image–text pairs (blue diagonal) while minimising it for non-matching pairs. Figure from [Radford et al. \(2021b\)](#).

2.4 Contrastive Learning and InfoNCE

Contrastive learning trains a model to produce representations in which similar inputs are close and dissimilar inputs are far apart. The InfoNCE loss [van den Oord et al. \(2018\)](#), a widely used contrastive objective, formalises this as a classification problem over a set of candidates.

Given a batch of B positive pairs $\{(\mathbf{h}_i, \mathbf{h}_i^+)\}_{i=1}^B$, where $\mathbf{h}_i$ and $\mathbf{h}_i^+$ are embeddings of two views of the same instance, the InfoNCE loss treats the matching pair as the positive class and all other $B - 1$ pairs in the batch as negatives:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{B} \sum_{i=1}^B \log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_i^+)/\tau)}{\sum_{j=1}^B \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j^+)/\tau)}, \quad (2.11)$$

where $\text{sim}(\cdot, \cdot)$ is a similarity function (typically cosine similarity) and $\tau > 0$ is a temperature parameter that controls the sharpness of the softmax distribution. A lower temperature makes the loss more sensitive to hard negatives by concentrating the probability mass on the highest-similarity candidates.

InfoNCE is closely related to the softmax cross-entropy loss: each sample i must “classify” its correct match $\mathbf{z}_i^+$ among B candidates. Minimising InfoNCE is equivalent to maximising a lower bound on the mutual information between the two views [van den Oord et al. \(2018\)](#).

In our pipeline, the two views are two scene graphs representing the same room (obtained either from different rescans or via subgraph augmentation), and cosine similarity is used as the similarity function. This objective learns an embedding space where scene graphs from the same room cluster together regardless of graph completeness, enabling generalisation to sparse text graphs at inference.

2.5 Determinantal Point Processes

A *determinantal point process* (DPP) [Kulesza and Taskar \(2012\)](#) is a probability distribution over the subsets of a finite ground set $\mathcal{Y} = \{1, \dots, N\}$ that explicitly favours *diverse* subsets: subsets containing items that are similar to one another receive lower probability than subsets made up of

distinct items. A DPP is parameterised by a real, symmetric, positive semi-definite kernel matrix $L \in \mathbb{R}^{N \times N}$, and the probability of drawing a subset $Y \subseteq \mathcal{Y}$ is proportional to the determinant of the corresponding principal submatrix:

$$\Pr(Y) \propto \det(L_Y), \quad (2.12)$$

where L_Y denotes the submatrix of L indexed by the elements of Y .

Geometrically, $\det(L_Y)$ equals the squared volume spanned by the rows of a matrix whose Gram matrix is L_Y . Adding an item that is very similar to one already in Y makes two rows nearly parallel, collapsing the volume to zero; conversely, items that are nearly orthogonal span a large volume and are therefore more likely to be selected together. This is the mechanism by which a DPP enforces diversity.

The kernel is conventionally decomposed as a *quality–similarity* factorisation

$$L_{ij} = q_i q_j S_{ij}, \quad (2.13)$$

where $q_i \geq 0$ is a scalar *quality* score for item i (how intrinsically desirable it is) and $S_{ij} \in [0, 1]$ is a *similarity* between items i and j . Under this form, L_Y balances two forces: increasing the q_i 's pushes the selected set towards individually high-quality items, while decreasing off-diagonal similarities spreads the selection across dissimilar items.

Finding the subset Y of maximum probability under (2.12) is NP-hard (Kulesza and Taskar, 2012), so in practice a greedy approximation is used. The map $Y \mapsto \log \det(L_Y)$ is a monotone submodular set function when L is positive semi-definite, and classical results on submodular maximisation then guarantee that greedy selection recovers a $(1 - 1/e)$ -approximation of the optimum under a cardinality constraint. *Greedy MAP-DPP* therefore starts from $Y = \emptyset$ and, at each step, adds the item that maximises the marginal gain

$$i^* = \arg \max_{i \notin Y} [\log \det(L_{Y \cup \{i\}}) - \log \det(L_Y)], \quad (2.14)$$

stopping when a target subset size k is reached or the marginal gain becomes non-positive. Each marginal gain can be evaluated in $O(k)$ time by maintaining a Cholesky factor of L_Y and updating it with a rank-one correction when an item is added, giving $O(Nk^2)$ total cost for the entire selection.

2.6 Geometric Foundations

This section reviews the geometric foundations that underpin the visibility reasoning used in fine localisation (Section 5.5).

2.6.1 Triangle Meshes

A triangle mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ is a piecewise-linear approximation of a surface, defined by a set of vertices $\mathcal{V} \subset \mathbb{R}^3$ and a set of triangular faces $\mathcal{F}$, where each face $f \in \mathcal{F}$ indexes three vertices. Triangle meshes are the dominant surface representation in computer graphics and 3D reconstruction because they offer an explicit, watertight boundary description that supports efficient geometric queries Botsch et al. (2010).

In indoor scene understanding, meshes are typically obtained from RGB-D reconstruction/S-LAM pipelines such as BundleFusion Dai et al. (2017b), ORB-SLAM Mur-Artal et al. (2015) or volumetric integration methods Newcombe et al. (2011). Each face carries a semantic label when instance segmentation is available, enabling queries of the form “which object does a given surface element belong to?”

2.6.2 Ray Casting

Ray casting is a fundamental operation in computational geometry that determines the intersection of a half-line (ray) with a set of geometric primitives [Glassner \(1989\)](#). Formally, given a ray origin $\mathbf{o} \in \mathbb{R}^3$ and a unit direction $\hat{\mathbf{r}} \in \mathbb{S}^2$, the ray is defined as the set of points

$$\mathbf{o} + t\hat{\mathbf{r}}, \quad t \geq 0.$$

A ray-casting query against a triangle mesh $\mathcal{M}$ returns the smallest $t^* > 0$ for which $\mathbf{o} + t^*\hat{\mathbf{r}}$ lies on a face of $\mathcal{M}$, together with the identity of that face. When no intersection exists, the ray is unoccluded along its entire extent.

For meshes with many faces, naïve ray-triangle testing is prohibitively expensive. Bounding volume hierarchies (BVHs), typically axis-aligned bounding box (AABB) trees, reduce the expected cost to $\mathcal{O}(\log |\mathcal{F}|)$ per query [Wald et al. \(2007\)](#). Modern libraries such as Open3D [Zhou et al. \(2018\)](#) expose GPU- or SIMD-accelerated BVH-based ray casters that handle millions of rays per second on commodity hardware.

2.6.3 Visibility Determination

An object is said to be *visible* from a viewpoint $\mathbf{o}$ if an unoccluded line of sight exists between $\mathbf{o}$ and some point on the object’s surface. In the point-to-centroid variant, a single ray is cast from $\mathbf{o}$ toward the object centroid μ_o ; the object is visible if the first mesh intersection lies on a face belonging to that object. More conservative multi-sample schemes cast rays toward several surface points and declare visibility when at least one ray is unoccluded.

Visibility reasoning has been widely used in robotics and computer vision for tasks such as view planning, next-best-view selection, and sensor placement. In the context of vision-language localisation, visibility provides a hard geometric constraint: a described object can appear in the observer’s percept only if it is visible from the observer’s position.

2.6.4 Field of View and Camera Geometry

A pinhole camera projects the 3D world onto a 2D image plane through a centre of projection. The *field of view* (FoV) defines the angular extent of the visible scene, typically parameterised by horizontal and vertical half-angles (α_h, α_v) [Hartley and Zisserman \(2003\)](#). An object whose direction from the camera centre falls outside the FoV cone cannot appear in the resulting image, regardless of whether it is geometrically unoccluded.

In localisation from natural-language descriptions, the FoV constraint captures the fact that all objects mentioned together in a description are likely to have been observed within a single view. Searching for the FoV window that maximises the number of simultaneously visible described objects therefore provides a strong cue for estimating the observer’s heading.

2.7 Bayesian Inference for Sequential Disambiguation

When an initial observation is ambiguous, several hypotheses may remain plausible. Bayesian inference provides a principled way to resolve this ambiguity by incorporating additional evidence sequentially. Let $\mathcal{H}$ denote a finite hypothesis set, and let $p_t(h)$ be the posterior probability of hypothesis $h \in \mathcal{H}$ after step t .

At each step, the system acquires an additional observation y_t . Given a likelihood model $\mathcal{L}(y_t | h)$, the posterior is updated by Bayes' rule:

$$p_t(h) = \frac{p_{t-1}(h) \mathcal{L}(y_t | h)}{\sum_{h' \in \mathcal{H}} p_{t-1}(h') \mathcal{L}(y_t | h')}. \quad (2.15)$$

This update increases the probability of hypotheses that are more consistent with the new observation and decreases the probability of those that are not.

The uncertainty of the current belief state can be quantified by the Shannon entropy:

$$H(p_t) = - \sum_{h \in \mathcal{H}} p_t(h) \log p_t(h). \quad (2.16)$$

As more informative observations are incorporated, the posterior typically becomes more concentrated and the entropy decreases. In sequential disambiguation problems, this process gradually narrows the set of plausible hypotheses until a confident decision can be made.

This Bayesian update framework is widely used in sequential state estimation, active perception, and interactive disambiguation, where uncertainty must be reduced over multiple steps rather than from a single observation.

Related Work

This chapter surveys the research areas that inform our pipeline: visual localization methods that define the dominant paradigm we depart from, language-based and cross-modal localization approaches that motivate our task, dialog-based systems for spatial disambiguation, and 3D scene graph representations that serve as the structural backbone of our method.

3.1 Visual Localization

Estimating the 6-DoF camera pose in a known environment is a long-standing problem in computer vision. We briefly review the main paradigms to contextualize our departure from image-based queries.

Structure-based methods build an explicit 3D map (typically a Structure-from-Motion point cloud) and localize by establishing 2D–3D correspondences between image features and map points, followed by a Perspective-n-Point (PnP) solver with RANSAC [Sattler et al. \(2017\)](#); [Taira et al. \(2018\)](#); [Sarlin et al. \(2019\)](#). Learned local features [DeTone et al. \(2018\)](#) and matching networks [Sarlin et al. \(2020\)](#) have substantially improved robustness to appearance changes. Hierarchical pipelines [Sarlin et al. \(2019\)](#) that first retrieve candidate database images and then perform local feature matching define the current dominant paradigm.

Pose regression methods take an alternative approach by training a CNN to directly regress the 6-DoF pose from a single image [Kendall et al. \(2015\)](#); [Brahmbhatt et al. \(2018\)](#). While architecturally simpler, these methods typically sacrifice accuracy compared to structure-based approaches. Scene coordinate regression methods [Brachmann et al. \(2017\)](#); [Brachmann and Rother \(2018\)](#); [Brachmann et al. \(2023\)](#) recover much of this accuracy gap by predicting dense 3D coordinates for each pixel and integrating a differentiable PnP+RANSAC solver, achieving state-of-the-art single-image localization.

Visual place recognition [Arandjelovic et al. \(2016\)](#); [Hausler et al. \(2021\)](#); [Berton et al. \(2022\)](#); [Ali-bey et al. \(2023\)](#) tackles the related retrieval problem of identifying the closest database location given a query image. This is analogous to the scene retrieval stage of our pipeline but uses an image rather than text as the query modality. Recent work has moved toward lighter map representations: **PixLoc** [Sarlin et al. \(2021\)](#) refines poses via learned feature-metric alignment, and **OrienterNet** [Sarlin et al. \(2023\)](#) localizes against 2D public maps.

All of the above methods require a visual query, an image or video stream, at inference time. This carries several drawbacks: bandwidth-heavy transmission (especially for indoor queries), potential privacy violations when photos capture sensitive surroundings, and the non-trivial requirement that users frame an informative shot. Our work departs from this paradigm entirely by replacing the image with a natural-language description, probing how much localization accuracy is attainable from language alone.

3.2 Language-based and Cross-modal Localization

A small but growing body of work explores localization from non-visual modalities, particularly natural language.

Text2Pos [Kolmet et al. \(2022\)](#) was the first to study text-to-3D localization. The method operates on large-scale outdoor point clouds from the KITTI-360 dataset: the environment is partitioned into coarse cells, and a contrastive learning framework trains paired text and point-cloud encoders to embed free-form descriptions and 3D cell representations into a shared space. At inference, the cell whose embedding is closest to the query embedding is retrieved. Text2Pos demonstrated that language alone can provide a meaningful spatial signal for localization, but its scope is limited in two important ways: it targets outdoor environments with relatively sparse geometry, and it performs only coarse cell-level retrieval without any mechanism for resolving a viewpoint or pose within the retrieved cell. Transferring the approach to cluttered indoor scenes, where object-level semantics dominate over geometric structure, is non-trivial: on the ScanScribe indoor benchmark, Text2Pos achieves only 11.6% Top-1 recall [Chen et al. \(2024\)](#).

In the indoor domain, [Chen et al. \(2024\)](#) frame the problem as scene retrieval over 3D scene graphs. A text query is first parsed into a text scene graph, a structured representation of mentioned objects and their spatial relations, and then matched against a database of 3DSSG [Wald et al. \(2020\)](#) scene graphs to identify the correct room. Their **Text2SGM** model explores three matching strategies (match probability, cosine similarity, and retrieval-based scoring) and introduces the ScanScribe benchmark, which pairs 4,472 natural-language descriptions with 3D scenes across 218 training and 55 test scenes. Text2SGM demonstrates that graph-based matching from language is feasible and achieves strong retrieval accuracy when query phrasing aligns with the training vocabulary. However, performance degrades sharply under domain shift (e.g. LLM-generated queries), and the method stops at coarse scene identification: the output is a scene label rather than a position and heading within that scene.

SceneGraphLoc [Miao et al. \(2024\)](#) performs cross-modal coarse localization by matching image features to 3D scene graphs. The architecture uses a dual-branch GATv2 encoder that processes geometric information (spatial displacements between objects) and semantic relational information (typed edges such as *left-of*, *on*) through separate branches, fusing them via a learned sigmoid gate that adapts per node. This design allows the encoder to exploit both spatial layout and semantic structure when available, while gracefully degrading when one modality is absent. We adapt this dual-branch design for text-to-graph retrieval, replacing the image branch with a text branch that operates on CLIP-embedded scene graphs parsed from natural language descriptions.

SGAligner [Sarkar et al. \(2023\)](#) tackles multi-modal scene-graph alignment by learning a shared embedding space across 3D point clouds, images, and text. The method uses a contrastive objective that aligns scene-graph representations from different modalities of the same scene while pushing apart representations from different scenes. SGAligner demonstrates that scene graphs provide a natural bridge between modalities, enabling cross-modal retrieval without requiring modality-specific architectures. However, like the methods above, SGAligner targets scene-level retrieval rather than fine-grained pose estimation within a retrieved scene.

Our work builds on these foundations but extends the pipeline beyond scene retrieval. The key contribution relative to this body of work is resolving a precise intra-scene pose from language, a problem that requires reasoning about viewpoint-dependent visibility and spatial configuration, which is not captured by global embedding similarity alone.

3.3 Dialog-based Localization and Embodied Navigation

In embodied settings, agents often interact with humans via dialog to resolve spatial ambiguity. This line of work motivates our dialog-based disambiguation module.

Vision-and-Dialog Navigation Thomason et al. (2020) introduces cooperative dialogs for object-goal navigation, where an agent asks clarification questions to narrow down possible target locations. **DiaLoc** Zhang et al. (2024) proposes iterative dialog-based localization, where an agent narrows its pose estimate through clarification questions grounded in visual observations. Both works demonstrate that interactive dialog can effectively reduce spatial ambiguity when a single observation is insufficient.

The broader **Vision-and-Language Navigation (VLN)** literature Anderson et al. (2018); Qi et al. (2020); Hong et al. (2021) studies instruction-following in photorealistic environments, where an agent must navigate to a goal location described in natural language. Recent LLM-based agents Zhou et al. (2024) bring explicit reasoning capabilities to navigation tasks, demonstrating that large language models can serve as planners for embodied agents.

A key challenge highlighted by this body of work is the inherent ambiguity of natural language: multiple spatial configurations may satisfy the same description, leading to multi-modal distributions over candidate poses. This motivates explicit uncertainty modelling, in our case a Bayesian posterior over candidate poses, rather than deterministic point estimation. Our dialog module differs from prior work in that it operates over a posterior distribution derived from 3D scene geometry and visibility, selecting questions to maximise expected information gain over this posterior.

3.4 3D Scene Graphs and Vision-Language Grounding

Scene graphs Armeni et al. (2019) encode objects, attributes, and spatial relationships in a unified hierarchical structure, providing a compact yet expressive representation of 3D environments. They serve as the structural backbone of our pipeline.

3D Semantic Scene Graphs (3DSSG) Wald et al. (2020) extend the scene graph concept to learned prediction of semantic graphs from 3D indoor reconstructions, annotating each object instance with semantic labels and pairwise spatial relations. Incremental methods such as **Scene-GraphFusion** Wu et al. (2021) enable on-the-fly graph construction from RGB-D streams. Open-vocabulary extensions using foundation models Gu et al. (2024); Peng et al. (2023); Kerr et al. (2023) have further broadened applicability by removing the need for a fixed vocabulary, though our current system operates on pre-defined label sets.

On the language side, several benchmarks pair referring expressions or questions with 3D scenes. **ScanRefer** (Chen et al., 2020) and **ReferIt3D** (Achlioptas et al., 2020) pair referring expressions with 3D bounding boxes, requiring models to ground a textual description to a specific object in a 3D scan. **ScanQA** (Azuma et al., 2022) and **SQA3D** (Ma et al., 2023) extend this to question answering with spatial reasoning, where the answer depends on understanding object arrangements within the scene. Pre-trained 3D-language models (Zhu et al., 2023; Hong et al., 2023) align point clouds or multi-view features with text at scale.

From a modelling perspective, scene graphs enable relational reasoning through graph neural networks, where message passing captures dependencies between objects and their spatial configuration Zhou et al. (2020). This is particularly important for localization, as viewpoint-

dependent descriptions often rely on relative object arrangements rather than absolute positions. Visual question answering (VQA) systems have leveraged similar ideas, constructing both image scene graphs and question graphs and applying GNNs to learn relational embeddings [Teney et al. \(2017\)](#). Some approaches condition graph construction on the input query for task-specific reasoning [Norcliffe-Brown et al. \(2018\)](#), while others incorporate external knowledge graphs [Wang et al. \(2018\)](#); [Narasimhan et al. \(2018\)](#).

Scene graph-based reasoning has also been applied to object detection, where GNNs refine region-of-interest features by modelling contextual relationships [Hu et al. \(2018\)](#); [Gu et al. \(2018\)](#), and to region classification, where relational structure guides semantic labelling [Chen et al. \(2018\)](#). These works highlight the importance of relational structure and message passing for reasoning over complex scenes.

However, all of the above resources target object grounding, retrieval, or question answering rather than *viewpoint-conditioned pose estimation*. This leaves a gap in both data and methods for fine-grained language-based localization that our work aims to fill: we use scene graphs not only for retrieval but also for visibility-based pose scoring and interactive dialog grounded in the graph structure.

3.4.1 Text Scene Graphs

A text scene graph $G^t = (V^t, E^t)$ is a structured representation of a natural language description, following the formalism of [Chen et al. \(2024\)](#). A free-form description is parsed into a directed graph whose nodes represent object mentions and whose edges represent explicitly stated spatial relations. Object mentions become nodes $v_i \in V^t$ labelled with their semantic class (e.g. `chair`, `table`), and a directed edge $(v_i, v_j) \in E^t$ is inserted if and only if the description explicitly states a spatial relation between object i and object j , carrying the relation string as its label (e.g. `next to`, `in front of`). No edges are inferred from co-occurrence alone. The resulting text graphs are intentionally sparse relative to the dense 3DSSG scene graphs they are matched against — a defining asymmetry of the cross-modal retrieval problem.

In this work, text graphs are provided by the ScanScribe dataset [Chen et al. \(2024\)](#) and our generated dataset (Section ??). Each description extracts objects, attributes, and spatial relations in JSON format, following the same procedure as [Chen et al. \(2024\)](#).

Data

Language-based indoor localization requires two kinds of data: 3D indoor scenes with semantic annotations from which scene graphs can be derived, and natural-language descriptions paired with those scenes (and, for fine-grained localization, with specific viewpoints inside them). This chapter describes both. Section 4.1 introduces the public benchmarks that we reuse from prior work (3RScan, ScanNet and ScanScribe), and Section 4.2 describes the new *LangLoc dataset*, a pose-indexed language benchmark we construct to support fine-grained localization, together with the processing pipeline used to generate it.

4.1 Benchmark Datasets

We build on three publicly available datasets. **3RScan** [Wald et al. \(2019\)](#) and **ScanNet** [Dai et al. \(2017a\)](#) provide RGB-D scans of indoor rooms together with reconstructed 3D meshes and per-vertex instance and semantic annotations. **3DSSG** [Wald et al. \(2020\)](#) augments a subset of 3RScan with object-level scene graphs, in which nodes correspond to object instances and directed edges encode typed spatial relations between them. **ScanScribe** [Chen et al. \(2024\)](#) pairs natural-language descriptions with 3DSSG scene graphs drawn from 3RScan and ScanNet, following a standard split of 218 scenes with 4,472 descriptions and 55 scenes with 1,116 descriptions.

4.2 The LangLoc Dataset

The public benchmarks above pair each description with a full scene, but not with a specific viewpoint inside that scene. Other 3D vision-language datasets are object-centric groundings [Chen et al. \(2020\)](#); [Achlioptas et al. \(2020\)](#) or scene-level captions [Zhu et al. \(2023\)](#), neither of which is suitable for fine-grained localization. To fill this gap we construct the *LangLoc dataset*, which extends 3RScan [Wald et al. \(2019\)](#) with *pose-indexed* egocentric descriptions: each description is associated with a specific camera pose inside a scene. The dataset covers 1,300+ indoor scans and 13,000+ natural-language descriptions.

The construction pipeline proceeds in four stages, detailed in the remainder of this section: image quality filtering, per-frame object-visibility computation, frame selection via a two-stage determinantal point process (DPP), and description generation from the selected frames.

4.2.1 Image Quality Assessment Filtering

Each 3RScan scene is captured as an RGB-D video: a sequence of image frames with associated depth maps and camera poses, recorded at roughly 30 Hz while the operator walks through the room. A typical scene yields several thousand such frames. Because these are consecutive video frames, neighbouring samples are highly redundant, and many are blurry, over- or under-exposed, or motion-corrupted. As a first-pass filter we apply a learned no-reference image quality model (QualiCLIP [Agnolucci et al. \(2025\)](#)): frames scoring below a dataset-dependent threshold in the range 0.4–0.5 are discarded before any geometric processing. To avoid over-pruning sparsely scanned scenes, we relax the threshold whenever fewer than 50 frames survive and instead retain the top-50 frames by quality score.

4.2.2 Visibility Computation via GPU Rasterisation

For every surviving frame we determine which annotated objects are visible by projecting the 3D mesh into the image using the frame’s known camera pose:

1. **Rasterisation.** The annotated mesh is rasterised into the frame with back-face culling, producing a per-pixel index of the triangle that is actually seen by that pixel.
2. **Per-face object IDs.** Each triangle is assigned the object instance label shared by the majority of its three vertices, so that the face-index buffer can be read as a per-pixel object-identity map.
3. **Filtering cascade.** An object is considered visible only if it passes all of:
 - Minimum pixel count (50–100 px depending on resolution),
 - Coverage threshold (2–5% of image area),
 - Depth consistency: the mesh-predicted depth z_{raster} must agree with the sensor-measured depth z_{sensor} within $|z_{\text{raster}} - z_{\text{sensor}}| < 0.2 \cdot z_{\text{sensor}}$. A mismatch means something other than the annotated mesh is actually in front of the camera (an occluder, a dynamic object, or a reconstruction error). The tolerance is relative because RGB-D noise grows with distance; 20% absorbs this noise without admitting true occluders.
 - Centroid in front of the camera and within depth range [0.2 m, 10.0 m].
4. **Output per object:** for each object that passes the filtering cascade, we record its semantic label, its centroid in both the world and the camera frame, the fraction of image pixels it covers, its distance from the camera, and the set of triangle faces that are actually visible in the frame. These per-frame visibility records are the primary input to the keyframe-selection and description-generation stages that follow.

4.2.3 Two-Stage DPP frame Selection

We select a compact set of high-quality, diverse frames per scene using a two-stage determinantal point process (DPP; see Section 2.5 for the necessary background). Recall that a DPP over a candidate set is parameterised by a kernel $L_{ij} = q_i q_j S_{ij}$, where q_i is a scalar quality score for item i and S_{ij} is a pairwise similarity; greedy MAP-DPP then picks items that maximise the determinant $\det(L_Y)$, trading off high quality against mutual diversity. In our setting each item is a candidate frame of the scene, and the two stages differ only in how q_i and S_{ij} are defined. Figure 4.1a illustrates the selection process.

Stage 1: Semantic DPP. The first stage selects up to $K_1 = 25$ semantically informative frames. For each candidate frame i , we compute a quality score:

$$q_i^{\text{base}} = H_v(i) \cdot S_{\text{rel}}(i) \cdot V_{\text{norm}}(i), \quad (4.1)$$

where H_v is the Shannon entropy over the visible-object pixel distribution (high when many similarly-sized objects are present) and $V_{\text{norm}} = 1 - \|\text{mean}(\hat{\mathbf{n}})\|$ is the normal variance over visible surface normals (high when geometrically diverse surfaces are observed). The relation density $S_{\text{rel}}(i)$ is the mean inverse pairwise distance between the centroids $\boldsymbol{\mu}_a$ of the n_i objects visible in frame i ,

$$S_{\text{rel}}(i) = \frac{1}{\binom{n_i}{2}} \sum_{1 \leq a < b \leq n_i} \frac{1}{\|\boldsymbol{\mu}_a - \boldsymbol{\mu}_b\|_2 + \varepsilon}, \quad (4.2)$$

then min-max normalised to $[0, 1]$ across all candidate frames of the scene. The small constant ε prevents division by zero when two objects are co-located.

Semantic similarity between frames uses CLIP [Radford et al. \(2021b\)](#) ViT-B/32 image embeddings $\boldsymbol{\psi}_i \in \mathbb{R}^{512}$, paired with the CLIP text encoder $\phi(\cdot)$ used elsewhere in the pipeline:

$$S_{ij}^{\text{sem}} = \frac{1}{2} (\cos(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j) + 1). \quad (4.3)$$

In Phase 1 (seeding), we construct the DPP kernel $L^{\text{seed}} = \mathbf{q}^{\text{base}} \mathbf{q}^{\text{base}^\top} \odot S^{\text{sem}}$ and select $k_{\text{seed}} = 2$ initial frames via greedy MAP-DPP with Cholesky updates. In Phase 2 (sequential expansion), vertex novelty is incorporated:

$$V_{\text{count}}(i) = \frac{1}{1 + \bar{c}(i)}, \quad q_i^{\text{full}} = q_i^{\text{base}} \cdot V_{\text{count}}(i), \quad (4.4)$$

where $\bar{c}(i)$ is the mean selection count of vertices visible in frame i . A conditioned greedy MAP-DPP iteratively picks one new frame that repels all previously selected frames, using q_i^{full} as the quality term. This repeats until K_1 frames are selected or the kernel becomes degenerate. No backfill is applied: quality is preferred over quota.

Stage 2: Spatial DPP. The second stage selects up to $K_2 = 10$ spatially diverse frames from the Stage-1 pool using a spatial similarity kernel. Let $\mathbf{c}_i^{\text{cam}} \in \mathbb{R}^3$ denote the 3D position of the camera centre of frame i and $\mathbf{v}_i \in \mathbb{S}^2$ its viewing direction. For a pair of frames (i, j) we define two scalar quantities local to this equation: the inter-camera distance $d_{ij}^{\text{cam}} = \|\mathbf{c}_i^{\text{cam}} - \mathbf{c}_j^{\text{cam}}\|_2 \in \mathbb{R}_{\geq 0}$ (in metres), and the angular difference between viewing directions $\Delta\theta_{ij} = \arccos(\mathbf{v}_i \cdot \mathbf{v}_j) \in [0, \pi]$ (in radians). The spatial similarity is

$$S_{ij}^{\text{spatial}} = \underbrace{\exp\left(-\frac{(d_{ij}^{\text{cam}})^2}{2\sigma_p^2}\right) \cdot \exp\left(-\frac{(\Delta\theta_{ij})^2}{2\sigma_a^2}\right)}_{S_{ij}^{\text{pose}}} \cdot \text{IoU}(i, j)^\gamma, \quad (4.5)$$

with position bandwidth $\sigma_p = 0.75$ m, angular bandwidth $\sigma_a = 20^\circ$, and IoU softening exponent $\gamma = 1.0$. The pose term S_{ij}^{pose} is 1 when two frames are at the same camera pose and decays smoothly as their positions or orientations differ; the IoU term is the pixel-space intersection-over-union of the two frames' labelled-object visibility masks, which captures how much actual scene content is shared between them. Their product makes S_{ij}^{spatial} small whenever either the viewpoints or the visible content differ, so the DPP will prefer frames that disagree on at least one of the two. When assembling the DPP kernel as in (2.13) we take uniform quality $q_i = 1$ for all frames, so the kernel reduces to $L_{ij}^{\text{spatial}} = S_{ij}^{\text{spatial}}$ and Stage 2 becomes a purely repulsive DPP

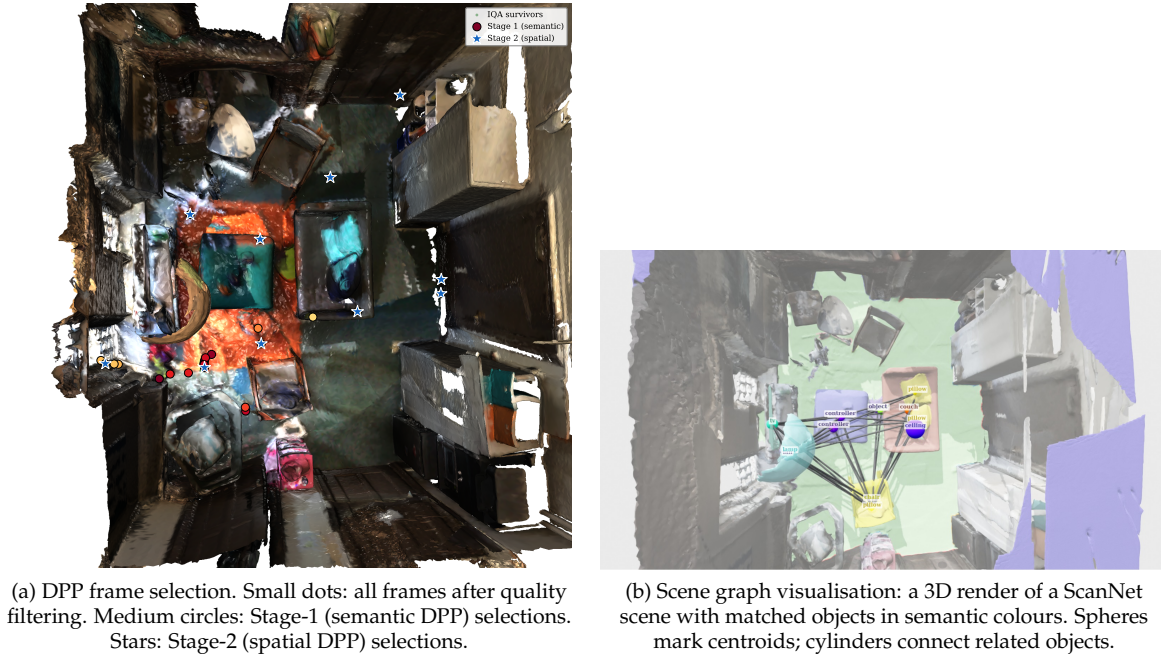


Figure 4.1: PER-SCENE OUTPUTS OF THE LANGLOC DATA PIPELINE. Subfigure (a) shows the frames selected by the two-stage DPP in a single scene: the quality-filtered candidate pool (dots), the semantic-DPP subset (circles), and the final spatially diverse subset (stars). Subfigure (b) shows the resulting 3D scene graph built from the reconstructed mesh for one of those scenes.

in which items compete on diversity alone. A one-shot greedy MAP-DPP selects K_2 frames; no backfill is applied.

On compact 3RScan rooms (typically 3–5 m across), the spatial DPP can be overly aggressive: with $\sigma_p = 0.75$ m, nearby viewpoints repel each other strongly, and the DPP may select fewer than K_2 frames before the principal submatrix L_Y^{spatial} loses rank. Concretely, when the candidate positions are clustered within a few bandwidths of one another, the off-diagonal entries of L^{spatial} are close to the diagonal entries, and the selected rows become nearly linearly dependent. The determinant $\det(L_{Y \cup \{i\}}^{\text{spatial}})$ then drops to (numerically) zero for every remaining candidate i , the marginal gain in (2.14) becomes non-positive, and greedy MAP-DPP terminates, exactly as in the stopping condition described in Section 2.5. This is by design: forcing diversity in a small room produces more informative descriptions. However, it means the final selected-frame count per scene varies (median: 8, range: 3–10 for 3RScan). For larger ScanNet++ scenes, the full $K_2 = 10$ quota is almost always reached.

4.2.4 Description Generation

Selected frames are captioned by GPT-4o-mini via the OpenAI API. For each frame, the visible object list and spatial relations (computed in camera coordinates) are assembled into a structured prompt. Symmetric and inverse relation duplicates (e.g. `left (A, B)` and `right (B, A)`) are deduplicated before prompting. The system prompt instructs the model to produce natural, conversational descriptions:

“You describe indoor camera views in natural, conversational language. Write as if you’re ca-

usually telling someone what you see. Weave spatial relationships naturally rather than stating them explicitly. Keep it very concise: 2–3 sentences maximum.”

4.2.5 Scene Graph Construction for ScanNet

For 3RScan scenes, pre-built 3DSSG (Wald et al., 2020) scene graphs are available. For ScanNet (Dai et al., 2017a) scenes, we construct scene graphs in the same format directly from the mesh segmentation: directed attributed graphs whose nodes and edges carry exactly the attributes described in Section 2.2. Throughout this construction the goal is to remain faithful to the 3DSSG schema, so that ScanNet scene graphs and the pre-built 3RScan ones are interchangeable downstream. This means we populate every relation category that 3DSSG defines, even when individual downstream stages later consume only a subset. The procedure is:

1. **Canonical room axes.** Directional predicates such as *left* or *front* only make sense relative to some chosen orientation of the room; using the world coordinate axes would tie them to whatever pose the scanner happened to be in when the scan was started. We therefore derive a coordinate frame that is intrinsic to the room itself, in three steps. First, the vertical axis $\mathbf{e}_{\text{up}}$ is fixed to the known gravity direction, so that “higher” and “lower” always agree with what a person standing in the room would call up and down. Second, we project all mesh vertices onto the floor (the plane orthogonal to gravity) and run principal component analysis on the resulting 2D point cloud; the eigenvector with the larger eigenvalue is the longest horizontal extent of the room, which we call the *front* axis $\mathbf{e}_{\text{front}}$. For a typical rectangular room this corresponds to the long wall direction. Third, the *right* axis is defined as $\mathbf{e}_{\text{right}} = \mathbf{e}_{\text{up}} \times \mathbf{e}_{\text{front}}$, completing a right-handed orthonormal basis $(\mathbf{e}_{\text{front}}, \mathbf{e}_{\text{right}}, \mathbf{e}_{\text{up}})$ attached to the room. Two scans of the same room taken from different starting poses will produce the same axes (up to a sign flip on the front axis, which is resolved by a deterministic convention), so any directional predicate computed in this frame is invariant to where the scanner started.
2. **Node attributes.** For each segmented object instance we record the attributes prescribed by the 3DSSG node format: a unique instance identifier, a semantic class label, and an oriented bounding box parameterised by its centroid $\boldsymbol{\mu}_i \in \mathbb{R}^3$, three orthonormal axes, and per-axis half-extents.
3. **Edge construction.** For every object pair whose centroid distance is below a threshold we emit the directed, typed predicates defined by the 3DSSG vocabulary, computed in the canonical room axes from step 1. We populate four relation families:
 - directional (*left*, *right*, *front*, *behind*, *higher than*, *lower than*),
 - proximity (*close by*),
 - support (*standing on*, *supported by*),
 - comparative (*bigger than*, *smaller than*, derived from bounding-box volume ratios).

The support and comparative families are not consumed by every downstream stage in our pipeline, but we still emit them so that the resulting graphs remain drop-in compatible with downstream code written against 3DSSG.

Following 3DSSG’s conventions, a small number of semantic filtering rules suppress predicates that are physically meaningless for certain object classes: *ceiling* only participates in *higher than*, *floor* only in *lower than* and *supported by*, and *walls* have a restricted set of directional relations. The maximum centroid distance threshold for emitting any predicate is 1.5 m for both 3RScan and ScanNet++ (see Figure 4.1b).

Approach

5.1 Problem Formulation

We propose a pipeline for language-based localization that not only identifies the correct scene but also estimates a precise intra-scene pose. Given a free-form textual query T and a database of 3D scenes $\mathcal{D}$, the goal is to estimate the observer pose as $\hat{\mathbf{p}} = (\hat{\mathbf{c}}, \hat{\boldsymbol{\theta}}) \in \mathbb{R}^2 \times \mathbb{S}^2$, where $\mathbf{c} \in \mathbb{R}^2$ denotes the 2D position on the floor plane and $\boldsymbol{\theta} \in \mathbb{S}^2$ denotes the direction vector.

3D scene database. Let $\mathcal{D} = \{\mathcal{S}_i\}_{i=1}^N$ denote a database of scanned indoor scenes. Each scene $\mathcal{S}_i$ can be represented as a 3D scene graph, which is formally defined as a directed attributed graph

$$G^s = (V^s, E^s), \quad (5.1)$$

where each node $v_i \in V^s$ represents a distinct object instance in the scene, and each directed edge $(v_i, v_j) \in E^s$ encodes a typed spatial relation between instance i and instance j .

Text scene graph. Following [Chen et al. \(2024\)](#), we parse T into a text scene graph $G^t = (V^t, E^t)$. Nodes represent objects; edges encode relational statements from the text. This enables structured matching against the database graphs $\{G_i^s\}$.

Tasks. Given $(T, \mathcal{D})$, the system executes three tasks:

1. **Scene retrieval.** The output of the graph neural network produces a ranked list of candidate scenes $\pi(T) = (i_1, \dots, i_K)$, where the ranking is induced by a scene matching score $s(G^t, G_i^s)$ over the database $\mathcal{D}$. Here, $s(\cdot, \cdot)$ denotes a scene matching score between the text scene graph and a candidate 3D scene graph.
2. **Fine localization.** For the top candidate scene, a 2D floor position $\hat{\mathbf{c}} \in \mathbb{R}^2$ and a direction vector $\hat{\boldsymbol{\theta}} \in \mathbb{S}^2$ is estimated. Consider the ground truth pose as $\mathbf{p}^* = (\mathbf{c}^*, \boldsymbol{\theta}^*)$, given the top-ranked scene $\mathcal{S}_{i_1}$, fine localization estimates the pose

$$\hat{\mathbf{p}} = (\hat{\mathbf{c}}, \hat{\boldsymbol{\theta}}) = \arg \max_{\mathbf{c} \in \mathbb{R}^2, \boldsymbol{\theta} \in \mathbb{S}^2} f(T, \mathcal{S}_{i_1}, \mathbf{c}, \boldsymbol{\theta}).$$

$f(\cdot)$ denotes a pose-level consistency/confidence score between the query T , the candidate scene $\mathcal{S}_{i_1}$, and a candidate pose $(\mathbf{c}, \boldsymbol{\theta})$.

3. **Dialog-based disambiguation.** When the posterior over poses is multi-modal, the system asks targeted yes/no questions about object presence or spatial relations to iteratively reduce ambiguity [Thomason et al. \(2020\)](#); [Zhang et al. \(2024\)](#).

5.2 Scene Representation

A core design decision in this work is how to represent a 3D indoor scene in a form that is simultaneously amenable to geometric reasoning and semantic matching against natural language. We adopt the *3D scene graph* as the canonical scene representation, a structured data format that encodes object-level semantic instances as nodes and their pairwise spatial relationships as typed edges. This section describes how such graphs are constructed for the two distinct operational contexts of the pipeline: (i) the *training-time* graphs, built programmatically from raw 3D point cloud data, and (ii) the *retrieval-time* database graphs, loaded directly from the 3DSSG dataset [Wald et al. \(2020\)](#).

5.2.1 3D Scene Graph Construction

Construction from point clouds. Scene graphs are constructed from a coloured point cloud and a corresponding semantic instance segmentation.

Instance extraction. The point cloud $\mathcal{P} = \{(\mathbf{x}_k, \mathbf{rgb}_k)\}_{k=1}^M$ is partitioned into n object instances by grouping points that share the same instance identifier. The point set belonging to instance i is denoted $\mathcal{P}_i \subset \mathcal{P}$, while M is the number of points in the point cloud.

Node attribute computation. Three quantities are derived per instance. The centroid $\boldsymbol{\mu}_i$ and mean RGB colour $\mathbf{k}_i$ are computed as per-point averages:

$$\boldsymbol{\mu}_i = \frac{1}{|\mathcal{P}_i|} \sum_{(\mathbf{x}, \cdot) \in \mathcal{P}_i} \mathbf{x} \in \mathbb{R}^3, \quad (5.2)$$

$$\mathbf{k}_i = \frac{1}{|\mathcal{P}_i|} \sum_{(\cdot, \mathbf{rgb}) \in \mathcal{P}_i} \mathbf{rgb} \in \mathbb{R}^3, \quad (5.3)$$

and the bounding radius d_i^{rad} is the maximum Euclidean distance from any point in the instance to its centroid,

$$d_i^{\text{rad}} = \max_{(\mathbf{x}, \cdot) \in \mathcal{P}_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2 \in \mathbb{R}. \quad (5.4)$$

The centroid $\boldsymbol{\mu}_i$ is stored in raw world-space metric coordinates; no global normalisation is applied at this stage. The axis-aligned bounding-box dimensions $\mathbf{b}_i \in \mathbb{R}^3$ are also computed from the per-axis extents of $\mathcal{P}_i$, but are used exclusively to derive spatial descriptors for unique instance labelling (Section 5.2.2) and do not enter the GATv2 node feature vector.

Node feature vector. The per-node feature vector concatenates the centroid, mean colour, and a CLIP [Radford et al. \(2021b\)](#) text embedding of the semantic label l_i :

$$\mathbf{f}_i = [\boldsymbol{\mu}_i \parallel \mathbf{k}_i \parallel \phi(l_i)] \in \mathbb{R}^{518}, \quad (5.5)$$

where $\phi(\cdot) \in \mathbb{R}^{512}$ is the CLIP ViT-B/32 text encoder. This 518-dimensional vector is the input to the GATv2 encoder (Section 5.4.1).

Edge construction via K -nearest neighbours. Edges are constructed geometrically. For each node v_i , the $K = 5$ spatially nearest nodes are found by Euclidean centroid distance:

$$\mathcal{N}(i) = \text{argtop-}K_{j \neq i} \left\{ -\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|_2 \right\}. \quad (5.6)$$

For each directed pair (i, j) with $j \in \mathcal{N}(i)$, a relational predicate is determined from directional and distance cues. Directional predicates (above, below, left_of, right_of, in_front_of, behind) are assigned by thresholding the signed components of $\Delta\boldsymbol{\mu}_{ij} = \boldsymbol{\mu}_j - \boldsymbol{\mu}_i$ against a direction threshold $\epsilon_{\text{dir}} = 0.20$ m. Distance predicates are derived from the inter-surface gap, with touching assigned when $\|\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\|_2 \leq 0.65(d_i^{\text{rad}} + d_j^{\text{rad}})$, near when $\|\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\|_2 < 0.80$ m, and close_by otherwise. When a touching predicate is active, directional predicates are restricted to the vertical axis (above/below) to suppress spurious lateral relations between abutting surfaces. Symmetric inverses (e.g. above $\leftrightarrow$ below, left_of $\leftrightarrow$ right_of) are inserted explicitly, and duplicate (j, i, r) triples are de-duplicated.

The geometric edge attribute vector for each edge (i, j) is assembled as

$$\mathbf{e}_{ij}^{\text{geom}} = [\Delta\boldsymbol{\mu}_{ij} \parallel \|\Delta\boldsymbol{\mu}_{ij}\|_2 \parallel d_i^{\text{rad}} \parallel d_j^{\text{rad}} \parallel \mathbf{0}_2] \in \mathbb{R}^8, \quad (5.7)$$

where $\Delta\boldsymbol{\mu}_{ij} = \boldsymbol{\mu}_j - \boldsymbol{\mu}_i \in \mathbb{R}^3$ is the relative displacement vector, $\|\Delta\boldsymbol{\mu}_{ij}\|_2 \in \mathbb{R}$ is its Euclidean norm, $d_i^{\text{rad}}, d_j^{\text{rad}} \in \mathbb{R}$ are the bounding radii (maximum point-to-centroid distance) of the two endpoint instances, and $\mathbf{0}_2 \in \mathbb{R}^2$ is a padding term that brings the total dimensionality to eight, matching the fixed input expectation of the GATv2 geometric branch. This vector encodes the relative displacement, Euclidean distance, and metric scale of both endpoint instances, giving the message-passing network full access to the local geometry of each edge without requiring a learned positional encoding.

5.2.2 Graph Preprocessing

Before a scene graph is passed to the encoder, three preprocessing steps are applied: label disambiguation via unique instance labelling, coordinate handling, and the rationale for the K -nearest-neighbour edge construction strategy.

Unique instance labelling. Raw semantic segmentation assigns a single class label, such as wall, chair, or table, to every instance of that category, regardless of how many such instances exist in the scene. A typical indoor reconstruction contains three to five wall segments, two or more chairs, and multiple instances of other common object classes. If all instances of the same class share the same label string l_i , their CLIP text embeddings $\phi(l_i)$ are identical by construction, collapsing all discriminative power for those nodes. The GATv2 encoder then receives node features that are indistinguishable for all same-class instances, making it impossible to learn spatial arrangements that depend on identifying *which* wall or chair is being referenced.

To resolve this, each instance is assigned a *unique label* that appends spatial qualifiers to the base class name. The qualifiers are derived from the instance centroid $\boldsymbol{\mu}_i$ relative to the scene bounding box $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}]$. The centroid is first normalised to $[0, 1]^3$ per axis solely for this computation:

$$\tilde{\mu}_i^{(x)} = \frac{\mu_i^{(x)} - x_{\min}}{x_{\max} - x_{\min}}, \quad \tilde{\mu}_i^{(y)} = \frac{\mu_i^{(y)} - y_{\min}}{y_{\max} - y_{\min}}, \quad \tilde{\mu}_i^{(z)} = \frac{\mu_i^{(z)} - z_{\min}}{z_{\max} - z_{\min}}. \quad (5.8)$$

A cardinal qualifier is assigned from the horizontal position using threshold $\epsilon_{\text{dir}} = 0.20$:

$$\text{cardinal}(i) = \begin{cases} \text{north} & \tilde{\mu}_i^{(y)} > 1 - \epsilon_{\text{dir}}, \\ \text{south} & \tilde{\mu}_i^{(y)} < \epsilon_{\text{dir}}, \\ \text{east} & \tilde{\mu}_i^{(x)} > 1 - \epsilon_{\text{dir}}, \\ \text{west} & \tilde{\mu}_i^{(x)} < \epsilon_{\text{dir}}, \\ \text{center} & \text{otherwise}, \end{cases} \quad (5.9)$$

and a height qualifier from the vertical position using threshold $\epsilon_{\text{ht}} = 0.15$:

$$\text{height}(i) = \begin{cases} \text{upper} & \tilde{\mu}_i^{(z)} > 1 - \epsilon_{\text{ht}}, \\ \text{lower} & \tilde{\mu}_i^{(z)} < \epsilon_{\text{ht}}, \\ \text{middle} & \text{otherwise}. \end{cases} \quad (5.10)$$

The unique label is then formed by concatenation:

$$l_i^{\text{unique}} = l_i^{\text{base}} _ \text{cardinal}(i) _ \text{height}(i), \quad (5.11)$$

so that three wall segments become, for example, `wall_north_upper`, `wall_south_lower`, and `wall_east_middle`. If two instances still share the same qualifier string after this procedure, the instance identifier i is appended as a final disambiguator. The unique label l_i^{unique} replaces the base label in all downstream CLIP embeddings, ensuring every node in the graph carries a distinct semantic-geometric descriptor.

Raw centroid coordinates in node features. The centroid normalisation is applied solely to derive the qualifier string and does not affect the features used by the model. The raw world-space centroid μ_i (5.2) is preserved and directly incorporated into the node feature vector, retaining the metric scale of the scene. No additional scene-level normalisation (e.g., centring or rescaling) is performed. This design is deliberate: the geometric branch of the GATv2 relies on relative displacements $\Delta\mu_{ij}$, which are inherently translation-invariant, while the absolute centroid μ_i is handled by the MLP encoder, allowing the network to learn positional cues without imposing hand-crafted normalisation.

Rationale for K -nearest-neighbour edges. KNN edges are available for any point cloud with an instance segmentation, making graph construction fully automatic and independent of any external relation annotation pipeline. This is important for scalability: the system can be trained on any RGB-D scan dataset without requiring manually or model-predicted semantic relations.

More subtly, training on KNN-derived predicates forces the model to generalise over relation *semantics* rather than memorise exact and specific predicate strings. Both KNN predicates (`left_of`, `above`, `near`) and database predicates (`left`, `front`, `close by`) are embedded with CLIP before being passed to the relation branch. Semantically equivalent predicates are therefore automatically close in embedding space regardless of surface form, where r denotes a relation (predicate) label and $\phi(r) \in \mathbb{R}^{512}$ is its CLIP embedding.:

$$\cos(\phi(r^{\text{KNN}}), \phi(r^{\text{db}})) \gg \cos(\phi(r^{\text{KNN}}), \phi(r^{\text{unrelated}})), \quad (5.12)$$

so the relation branch generalises from training-time predicates to database predicates without any explicit vocabulary alignment step. The neighbourhood size $K = 5$ is chosen to balance graph connectivity (too small a K yields disconnected subgraphs in sparse scenes) against computational cost and over-smoothing, where a large K forces the GATv2 to aggregate over geometrically distant and semantically irrelevant neighbours.

5.3 Text Representation

At retrieval time, a natural language query T must be converted into a structured representation compatible with the graph encoder. To do this, text queries are represented as directed graphs $G^t = (V^t, E^t)$ parsed from natural language descriptions; the construction of these graphs is detailed in Section 3.4.1. Here we describe how text graph nodes and edges are encoded into the shared feature space used by the retrieval encoder.

It is important to state that text graphs are not used during training at any point, and only used in retrieval. Instead, training relies entirely on pairs of 3D scene graphs constructed from the same 218 scene IDs that constitute the ScanScribe training split (Section 5.4.4). This means the model never observes a text graph during optimisation; at inference it generalises to text graph queries solely through the shared 518-dimensional feature space and zero-padding scheme described in Section 5.3.2. The alignment between the two modalities is therefore an emergent property of the shared encoder architecture rather than the result of explicit cross-modal supervision.

5.3.1 Text-to-Graph Conversion

Text graphs are provided by the ScanScribe dataset [Zhu et al. \(2023\)](#) and our generated dataset (Chapter 4) consumed directly by our pipeline; we perform a similar parsing step as [Chen et al. \(2024\)](#), each description extracts objects, attributes and spatial relations in JSON format. Object mentions become nodes $v_i \in V^t$ labelled with their semantic class (e.g. `chair`, `table`), and a directed edge $(v_i, v_j) \in E^t$ is inserted if and only if the description explicitly states a spatial relation between object i and object j , carrying the relation string as its label (e.g. `next to`, `in front of`). No edges are inferred from co-occurrence alone. The resulting text graphs are intentionally sparse relative to the dense 3DSSG scene graphs they are matched against, a defining asymmetry of the cross-modal retrieval problem.

5.3.2 Feature Encoding

Node features. Text graph nodes are encoded into the same 518-dimensional feature vector as 3D scene graph nodes (5.5), ensuring the encoder operates in a unified feature space across modalities. Since text descriptions carry no spatial or colour information, the centroid fields are set to zero and the colour fields to a neutral grey:

$$\mathbf{f}_i^{\text{text}} = \mathbf{0}_3 \parallel \mathbf{c}_{\text{grey}} \parallel \varrho(l_i) \in \mathbb{R}^{518}, \quad (5.13)$$

where $\mathbf{c}_{\text{grey}} = (0.502, 0.502, 0.502)$ is a constant neutral colour and $\varrho(l_i) \in \mathbb{R}^{512}$ is the CLIP ViT-B/32 embedding of the node label string.

Because all text node centroids are $\mathbf{0}_3$, every pairwise displacement $\Delta\boldsymbol{\mu}_{ij} = \boldsymbol{\mu}_j - \boldsymbol{\mu}_i = \mathbf{0}$ and the bounding radii are a uniform constant $d^{\text{rad}} = 0.4$ m. The geometric edge attributes (5.7) therefore carry no discriminative spatial signal for text graph edges, which naturally suppresses the geometric branch contribution. The learned gate α_i (5.21) consequently shifts weight toward the relation branch, which is the only branch that carries meaningful signal for a text query.

The 518-dimensional feature space retains centroid and colour fields even for text nodes, as human descriptions naturally include spatial and colour cues (*“the red chair in the corner”*). The current benchmark dataset with spatially grounded text descriptions does not provide this for every node, so these fields default to $\mathbf{0}_3$ and neutral grey respectively. However, preserving the shared feature space is what enables zero-shot transfer from 3D training to text queries at inference — a reduced text-only feature dimension would require separate encoder weights and break the unified embedding space.

Edge features. Every edge from both graph modalities is encoded by passing its relation string through the CLIP text encoder:

$$\mathbf{e}_{ij}^{\text{rel}} = \phi(r_{ij}) \in \mathbb{R}^{512}, \quad (5.14)$$

where r_{ij} is the relation label.

Scene-level descriptor. A global CLIP descriptor is computed for each graph by encoding a sentence that lists all unique object labels:

$$\mathbf{u}(G) = \phi(\text{"A room with } l_1, \dots, l_K") \in \mathbb{R}^{512}, \quad \{l_k\}_{k=1}^K = \mathcal{L}(G), \quad (5.15)$$

where $\mathcal{L}(G)$ is the set of unique object labels in G . This holistic bag-of-objects summary complements the structural node embeddings and enters both the final graph embedding (Section 5.4.2) and the retrieval scoring function (Section 5.4.3).

5.4 Cross-modal Retrieval Model

Given a text graph G^t and a database of 3D scene graphs $\{G_i^s\}_{i=1}^N$, the retrieval model maps both modalities into a shared embedding space and ranks scenes by similarity. The architecture consists of a dual-branch graph encoder (Section 5.4.1), a graph-level pooling and projection step (Section 5.4.2), a weighted inference scoring function (Section 5.4.3), and a contrastive training objective (Section 5.4.4).

5.4.1 Graph Encoder Architecture

Graph encoder. We now describe the graph encoder used in our model. The encoder maps an input graph with node features and edge attributes to contextualised node embeddings using a

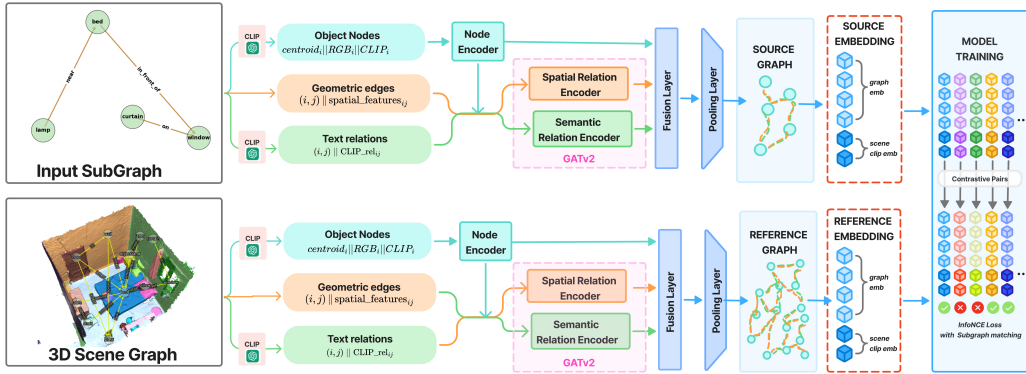


Figure 5.1: COARSE RETRIEVAL PIPELINE OVERVIEW. Given a natural language query, we first construct a scene graph representation where nodes encode object-level features and edges capture relational constraints. This graph is processed through a dual-branch architecture that maintains separate pathways for geometric and semantic information before adaptive fusion. During training, an InfoNCE objective learns to align text-derived query graphs with 3D database graphs by learning discriminative embeddings. This simultaneously pulls together matching query-database pairs while pushing apart non-matching pairs.

dual-branch GATv2 architecture. The encoder maps any graph G , whether a 3D scene graph or a text graph, to a set of node embeddings $\{\mathbf{h}_i\}$, where $\mathbf{h}_i^{(0)}$ denotes the initial node embedding obtained by projecting the input feature vector $\mathbf{f}_i$ into the d -dimensional latent space. The encoder consists of two parallel branches that process geometric and semantic relational information separately before fusing them via a learned gate.

Node projection. Each node feature vector $\mathbf{f}_i \in \mathbb{R}^{518}$ is first projected to the working dimension $d = 256$ by a shared MLP:

$$\mathbf{h}_i^{(0)} = \text{MLP}(\mathbf{f}_i) \in \mathbb{R}^d. \quad (5.16)$$

Graph attention layers. Each GATv2 layer updates node representations via attention-based message passing. For a node i , the updated representation is computed as:

$$\mathbf{h}_i^{(l+1)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \mathbf{W} \mathbf{h}_j^{(l)}, \quad (5.17)$$

where α_{ij} are attention coefficients computed as

$$\alpha_{ij} = \text{softmax}_j \left(a(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \mathbf{e}_{ij}) \right), \quad (5.18)$$

with $a(\cdot)$ a learnable attention function that incorporates both node features and edge attributes.

Geometric branch. The geometric branch constructs a K -nearest-neighbour graph ($K = 5$) over node centroids and propagates messages conditioned on the geometric edge attribute $\mathbf{e}_{ij}^{\text{geom}} \in \mathbb{R}^8$ (5.7). Two GATv2 layers (Brody et al., 2022) are stacked, each with 2 attention heads, $d = 256$ input and output channels, a residual connection, and GELU activation. Each layer computes attention scores jointly over node and edge features — the key property of GATv2 over the original GAT, which decouples the attention mechanism from the static linear transformation. A single layer normalisation is applied after the full stack:

$$\mathbf{g}_i = \text{LayerNorm} \left(\text{GATv2}_{\text{geom}}^{(2)} \left(\{\mathbf{h}_j^{(0)}, \mathbf{e}_{ij}^{\text{geom}}\}_{j \in \mathcal{N}(i)} \right) \right) \in \mathbb{R}^d, \quad (5.19)$$

where $\mathcal{N}(i)$ denotes the $K = 5$ nearest spatial neighbours of node i , and dropout with $p = 0.1$ is applied within each layer during training.

Relation branch. The relation branch operates on the typed edges E of the graph, embedding each relation string with CLIP before message passing. Two GATv2 layers are stacked with the same configuration as the geometric branch, with a single layer normalisation applied after the full stack:

$$\mathbf{t}_i = \text{LayerNorm} \left(\text{GATv2}_{\text{rel}}^{(2)} \left(\{\mathbf{h}_j^{(0)}, \mathbf{e}_{ij}^{\text{rel}}\}_{j: (i,j) \in E} \right) \right) \in \mathbb{R}^d, \quad (5.20)$$

where $\mathbf{e}_{ij}^{\text{rel}} = \phi(r_{ij}) \in \mathbb{R}^{512}$ is the CLIP embedding of relation label r_{ij} (5.14). The relation branch is the primary source of signal for text graph nodes, as it does not depend on geometric information.

Gated fusion. The outputs of the geometric and relation branches are fused per node via a learned gate $\alpha_i \in (0, 1)^d$:

$$\alpha_i = \sigma(W[\mathbf{g}_i \parallel \mathbf{t}_i]), \quad (5.21)$$

$$\tilde{\mathbf{u}}_i = \alpha_i \odot \mathbf{g}_i + (1 - \alpha_i) \odot \mathbf{t}_i + 0.1 \cdot \text{MLP}([\mathbf{g}_i \parallel \mathbf{t}_i]), \quad (5.22)$$

$$\mathbf{u}_i = \text{LayerNorm}(\tilde{\mathbf{u}}_i), \quad (5.23)$$

where $W \in \mathbb{R}^{d \times 2d}$ is a learned linear projection, $\sigma(\cdot)$ is the sigmoid function, and $\odot$ denotes element-wise multiplication. The MLP applies LayerNorm, a linear layer, GELU activation, dropout ($p = 0.1$), and a final linear layer to the concatenated branch outputs, providing a learned residual correction weighted at 0.1. The gate is learned end-to-end and adapts per node: for 3D scene graph nodes with full geometric information, α_i can place weight on either branch, whereas for text graph nodes with zero-padded geometry, the gate learns to rely primarily on $\mathbf{t}_i$. During training, a small Gaussian noise term $\epsilon \sim \mathcal{N}(\mathbf{0}, 0.01 \mathbf{I})$ is added to $\mathbf{u}_i$ as a regulariser.

Handling the label asymmetry. The unique instance labels introduced in Section 5.2.2 are applied only to 3D scene graph nodes; text graph nodes retain their plain semantic labels as extracted from the description. This asymmetry is deliberate. In 3D scene graphs, unique labels are necessary because multiple instances of the same class (e.g. three walls) would otherwise produce identical CLIP embeddings, collapsing all discriminative power for those nodes. Text graphs do not require this mechanism because natural-language descriptions disambiguate co-class instances through relational context (e.g. “the chair next to the window”) rather than through spatial qualifiers. The cross-modal gap introduced by this asymmetry is small in practice: because CLIP processes the unique label as a compositional phrase in which the base noun dominates, the cosine similarity between a plain label embedding $\phi(\text{chair})$ and a qualified label embedding $\phi(\text{chair_north_upper})$ remains high. Any residual distributional shift is absorbed by the gated fusion mechanism (5.21), which already learns modality-dependent weighting through the zero-padded geometry signal.

5.4.2 Graph-level Embedding

The node embeddings $\{\mathbf{h}_i\}$ are aggregated into a single graph-level vector by mean pooling, then concatenated with the scene-level CLIP descriptor $\mathbf{s}(G) \in \mathbb{R}^{512}$ (5.15) and passed through a fusion MLP. Mean pooling is permutation-invariant, treating all nodes equally regardless of graph size. This is well-suited to the small scene graphs in this setting (typically 10–30 nodes per scene), where the difference between a weighted and unweighted sum is marginal. The concatenation of $\mathbf{s}(G)$ injects a holistic bag-of-objects signal not captured by the structural node embeddings alone, providing complementary coarse-grained information about scene composition. The fusion MLP learns both signals from the node embedding and the scene description independently, then applies LayerNorm, a linear projection with ReLU activation, dropout with rate 0.3, a second linear projection, and a final LayerNorm, yielding the graph embedding:

$$\mathbf{z}(G) = \text{MLP}_{\text{fusion}} \left(\left[\frac{1}{|V|} \sum_i \mathbf{h}_i \parallel \mathbf{s}(G) \right] \right) \in \mathbb{R}^d. \quad (5.24)$$

5.4.3 Similarity Function

At inference, each database scene G_i^s is scored against the query graph G^t by a weighted combination of three complementary signals:

$$\text{score}(G^t, G_i^s) = w_{\text{emb}} \cos(\mathbf{z}(G^t), \mathbf{z}(G_i^s)) + w_{\text{glob}} \cos(\mathbf{u}(G^t), \mathbf{u}(G_i^s)) + w_{\text{jac}} \text{overlap}(\mathcal{L}(G^t), \mathcal{L}(G_i^s)), \quad (5.25)$$

where $\text{overlap}(\mathcal{L}^t, \mathcal{L}^s)$ is the Sørensen–Dice similarity between the sets of unique node labels,

$$\text{overlap}(\mathcal{L}^t, \mathcal{L}^s) = \frac{2 |\mathcal{L}^t \cap \mathcal{L}^s|}{|\mathcal{L}^t| + |\mathcal{L}^s|}, \quad (5.26)$$

ranging from 0 (no shared labels) to 1 (identical label sets).

The three terms capture complementary aspects of scene–query compatibility. The embedding similarity $\cos(\mathbf{z}(G^t), \mathbf{z}(G_i^s))$ captures structural and relational similarity learned by the encoder. The global CLIP similarity $\cos(\mathbf{u}(G^t), \mathbf{u}(G_i^s))$ provides a coarse holistic check on scene composition. Although $\mathbf{u}(G)$ is already incorporated into $\mathbf{z}(G)$ via the fusion MLP (5.24), the learned projection may attenuate this signal during training. The direct term $\cos(\mathbf{u}(G^t), \mathbf{u}(G_i^s))$ provides an unsuppressable bypass that guarantees the holistic scene composition signal contributes to ranking independently of the encoder. The label F1 term provides a lexical grounding signal that is robust when the encoder embedding is uncertain. The weights w_{emb} , w_{glob} , and w_{jac} are tuned on the validation split. Scenes are ranked in descending order of $\text{score}(G^t, G_i^s)$ to produce the retrieved list $\pi(T)$:

$$\pi(T) = \text{argsort}_i \left\{ -\text{score}(G^t, G_i^s) \right\}, \quad (5.27)$$

where $\text{score}(G^t, G_i^s)$ is defined in (5.25). The resulting ranked list is used to evaluate coarse localization performance, measuring whether the correct scene is retrieved within the top- k predictions (Top- k accuracy), as reported in Section 6.

5.4.4 Training Objective

The encoder is trained using positive pairs of 3D scene graphs drawn from the same scene, with negatives sampled randomly from other scenes in the batch. To reiterate, no text graphs are used during training; the model learns purely from 3D scene graph self-supervision and generalises to text graph queries at inference through the shared 518-dimensional feature space and zero-padding scheme described in Section 5.3.2.

Positive pair construction. A positive pair (G_i^s, G_i^{s+}) consists of two scene graphs that describe the same physical room. Where available, the two graphs are drawn from different rescans of the same room, identified by a shared room ID in the 3RScan metadata. Using rescans as positives is a deliberate design choice: two rescans of the same room may differ in object placement, viewpoint, scan density, and even object inventory (due to scene changes between capture sessions), making the matching task genuinely challenging and preventing the encoder from exploiting low-level artefacts. When a room has no available rescan, the same scene graph is used twice with independent random subgraph augmentations as a fallback positive pair.

Subgraph augmentation. Each scene graph in a training pair is independently subsampled before being passed to the encoder. With probability 0.5 per sample, a random subset of $p \in [0.40, 0.70]$ of the nodes is retained, along with all edges between retained nodes; the remaining nodes and their incident edges are discarded. This stochastic subgraph augmentation serves two purposes. First, it acts as a regulariser, preventing the encoder from overfitting to the full graph

structure of any particular scene. Second, and more importantly, it directly mirrors the partial-observation nature of text queries: a natural language description typically mentions only a small subset of the objects present in a room, so training on partial graphs teaches the encoder to match incomplete observations against full scene graphs in the database. Let $\tilde{G}_i$ denote a randomly subsampled version of G_i . The encoder is trained such that two independently subsampled views of the same scene are closer in embedding space than views from different scenes:

$$\cos(\mathbf{z}(\tilde{G}_i^{(1)}), \mathbf{z}(\tilde{G}_i^{(2)})) \gg \cos(\mathbf{z}(\tilde{G}_i^{(1)}), \mathbf{z}(\tilde{G}_j^{(2)})), \quad \forall j \neq i. \quad (5.28)$$

Optimisation. The encoder is trained for 100 epochs using AdamW [Loshchilov and Hutter \(2019\)](#) with learning rate 10^{-3} , weight decay 10^{-4} , and gradient clipping with maximum norm 1.0.

5.5 Fine Localization

Given the ranked scene list $\pi(T) = (i_1, \dots, i_K)$ from the retrieval stage, fine localization operates on the top-ranked scene $\mathcal{S}_{i_1}$ with scene graph $G_{i_1}^s = (V_{i_1}^s, E_{i_1}^s)$. The goal is to estimate the observer pose

$$\hat{\mathbf{p}} = (\hat{\mathbf{c}}, \hat{\boldsymbol{\theta}}) \in \mathbb{R}^2 \times \mathbb{S}^2, \quad (5.29)$$

where $\hat{\mathbf{c}}$ is the 2D floor position and $\hat{\boldsymbol{\theta}}$ is the viewing direction vector. Figure 5.2 provides an overview of the pipeline.

Our key assumption is that a correct pose must jointly satisfy two conditions: it should lie at a location from which the objects mentioned in the query are visible, and it should admit a viewing direction under which these objects are simultaneously consistent with a realistic camera field of view. Based on this intuition, the method first establishes correspondences between textual entities in G^t and object instances in $G_{i_1}^s$ (Section 5.5.1). It then samples candidate observer positions on the floor plane (Section 5.5.2) and determines, via ray casting against the scene mesh, which matched objects are visible from each candidate (Section 5.5.3). A FoV-consistent viewing direction is estimated and used to score every candidate (Sections 5.5.4 and 5.5.5). Finally, the scored pose–direction hypotheses are aggregated into the final estimate (Section 5.5.6).

5.5.1 Text-to-Scene Object Matching

We first establish correspondences between the query graph $G^t = (V^t, E^t)$ and the retrieved scene graph $G_{i_1}^s$. Each query node and scene node is represented by its semantic label, which we embed into $\mathbb{R}^d$ using CLIP ([Radford et al., 2021a](#)). Let

$$\mathbf{E}^t \in \mathbb{R}^{|V^t| \times d}, \quad \mathbf{E}^s \in \mathbb{R}^{|V_{i_1}^s| \times d}$$

denote the matrices of ℓ_2 -normalised query-node and scene-node embeddings, respectively. We then compute the cosine similarity matrix

$$\mathbf{A} = \mathbf{E}^t (\mathbf{E}^s)^\top \in \mathbb{R}^{|V^t| \times |V_{i_1}^s|}.$$

To obtain a one-to-one matching, query nodes are greedily assigned to their best still-unused scene nodes, subject to a minimum similarity threshold. This uniqueness constraint prevents multiple textual entities from collapsing onto the same scene object. In practice, we optionally cap the number of retained matches (query nodes $\leftrightarrow$ scene nodes) by a nominal top- K budget and

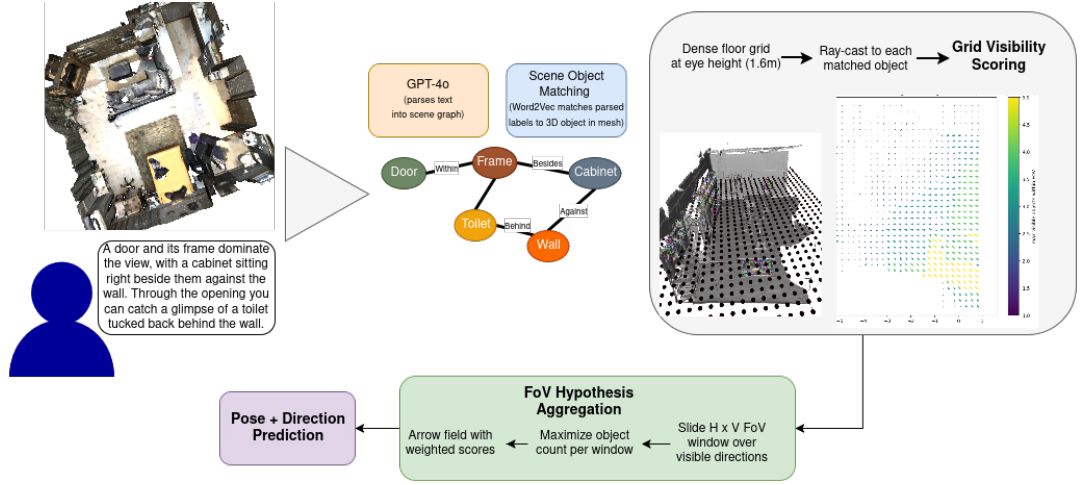


Figure 5.2: FINE-LOCALIZATION PIPELINE. Given a text scene graph G^t of objects and spatial relations matched to 3D object instances in the scene mesh (cf. Figure 4.1b). A dense floor grid of candidate positions is generated at eye height $h = 1.6$ m. From each candidate, omnidirectional ray casting against the mesh determines which matched objects are visible, independently of any assumed viewing angle. A FoV-consistent sweep then identifies the camera heading $\hat{\theta}$ that captures the largest subset of visible objects, yielding a scored arrow field from which the final pose $(\hat{c}, \hat{\theta})$ is estimated.

adapt this budget to the size of the query graph, which encourages broad coverage of the entities mentioned in the description rather than repeatedly selecting a few high-frequency objects.

Let $\mathcal{O}$ denote the set of matched scene objects retained after this stage, and let $\mu_o \in \mathbb{R}^3$ be the 3D centroid of object $o \in \mathcal{O}$. These matched objects are the anchors used for downstream geometric reasoning.

Note that scene-graph nodes carry the unique instance labels introduced in Section 5.2.2 (e.g. `chair_north_upper`), whereas text-graph nodes retain their plain semantic labels (e.g. `chair`). As discussed in Section 5.4.1, this asymmetry does not impair matching: CLIP processes the qualified label as a compositional phrase in which the base noun dominates, so cross-modal cosine similarity remains high.

5.5.2 Candidate Floor Positions and Inside-Scene Filtering

We discretise the walkable floor region of $\mathcal{S}_{i1}$ with a uniform grid

$$\mathcal{C} = \{\mathbf{c}_k\}_{k=1}^{K_c}, \quad \mathbf{c}_k \in \mathbb{R}^2,$$

using spacing Δ . Each $\mathbf{c}_k$ is a candidate observer position on the floor plane. For ray-casting, the omnidirectional visibility test described in Section 5.5.3, we lift each candidate to a 3D eye position

$$\tilde{\mathbf{c}}_k = [\mathbf{c}_k^\top, h]^\top \in \mathbb{R}^3,$$

where h is a fixed eye height.

Not all grid samples correspond to feasible positions inside the scene. We therefore prune invalid candidates using a downward ray-casting test on the reconstructed mesh. A candidate is retained only if a ray cast along the negative vertical direction intersects the floor within a small distance below the eye point. This removes grid cells outside the room or above non-walkable

regions, preventing spurious visibility scores near the scene boundary. Note that no viewing direction is assumed at this stage: the subsequent visibility test (Section 5.5.3) casts rays from each eye point toward every matched object centroid independently of camera orientation, effectively treating the observer as an omnidirectional sensor. The viewing direction is determined only afterward, in Section 5.5.4, by searching for the field-of-view window that captures the largest subset of visible objects. The downward ray cast used here for inside-scene filtering is similarly direction-free — it tests only whether the candidate position lies above a valid floor surface.

5.5.3 Ray-Cast Visibility Scoring

For each retained candidate position $\mathbf{c}_k$ and each matched object $o \in \mathcal{O}$ with 3D centroid $\boldsymbol{\mu}_o$, we cast a ray from the eye point $\tilde{\mathbf{c}}_k$ toward $\boldsymbol{\mu}_o$ using the `Open3D RaycastingScene` built from the reconstructed triangle mesh $\mathcal{M}$ of $\mathcal{S}_{i_1}$. The ray direction is

$$\hat{\mathbf{r}}_{ko} = \frac{\boldsymbol{\mu}_o - \tilde{\mathbf{c}}_k}{\|\boldsymbol{\mu}_o - \tilde{\mathbf{c}}_k\|_2}.$$

Object o is deemed *visible* from $\mathbf{c}_k$ if the first intersection of this ray with $\mathcal{M}$ lies on a triangle belonging to o , i.e. no intervening geometry (walls, furniture, etc.) occludes the line of sight.

The underlying intuition is that the observer must stand at a position from which the objects mentioned in the description are simultaneously visible and spatially nearby. By evaluating visibility across the dense grid $\mathcal{C}$, we obtain, for every candidate, the subset of matched objects that are directly observable. This per-position visibility set is the input to the subsequent FoV-consistency and scoring stages.

5.5.4 Visibility Reasoning and FoV-Consistent Direction Estimation

Given the per-position visibility sets computed in Section 5.5.3, we now estimate a coherent viewing direction for each candidate. For every object o visible from $\mathbf{c}_k$, we reuse the ray direction $\hat{\mathbf{r}}_{ko}$ defined in Section 5.5.3 — now interpreted as a candidate viewing direction rather than a visibility query:

$$\hat{\mathbf{r}}_{ko} = \frac{\boldsymbol{\mu}_o - \tilde{\mathbf{c}}_k}{\|\boldsymbol{\mu}_o - \tilde{\mathbf{c}}_k\|_2} \in \mathbb{S}^2.$$

Each vector $\hat{\mathbf{r}}_{ko}$ describes the direction to a single object, but the text query implies that multiple objects were observed together in one view. A correct pose must therefore not only see the right objects individually, but also admit a single camera orientation under which they fall within a common field of view. To bridge from per-object directions to a coherent multi-object viewing direction, we search over candidate FoV windows for the one that simultaneously contains the largest subset of the per-object direction vectors $\{\hat{\mathbf{r}}_{ko}\}_{o \in \mathcal{O}}$.

Concretely, we convert each $\hat{\mathbf{r}}_{ko}$ to yaw–pitch coordinates and sweep an axis-aligned horizontal-FoV $\times$ vertical-FoV window across the resulting angular point set. Because the yaw domain is periodic, a naïve linear sweep would miss windows that straddle the $\pm\pi$ boundary. Rather than introducing a circular distance metric — which would prevent the use of a simple sorted-order sweep — we duplicate all yaw values with a 2π offset and sweep the extended linear sequence, guaranteeing that every contiguous angular interval is considered exactly once at the cost of doubling the sweep length. For each candidate horizontal window, a second one-dimensional sweep over pitch finds the best vertical interval.

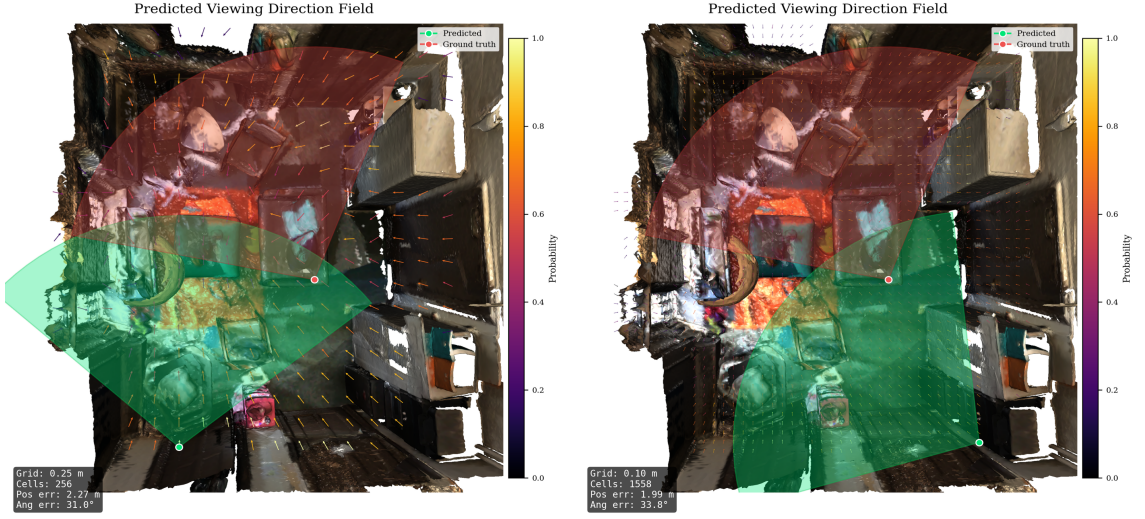


Figure 5.3: DIRECTION FIELD. Predicted viewing direction at each grid cell overlaid on the top-down view at two resolutions: coarse grid ($\Delta = 0.25$ m, left) and fine grid ($\Delta = 0.10$ m, right). Arrow colour encodes probability (inferno colourmap: dark = low, bright = high). The predicted FoV wedge covers the region described by the query text.

Let $\mathcal{O}_k^{\text{FoV}} \subseteq \mathcal{O}$ denote the subset of matched objects supported by the optimal FoV window at position $\mathbf{c}_k$. The associated candidate viewing direction is then

$$\hat{\theta}_k = \frac{\sum_{o \in \mathcal{O}_k^{\text{FoV}}} \hat{\mathbf{r}}_{ko}}{\left\| \sum_{o \in \mathcal{O}_k^{\text{FoV}}} \hat{\mathbf{r}}_{ko} \right\|_2} \in \mathbb{S}^2.$$

To reduce computational cost, this FoV search is evaluated only on a subsampled set of grid positions controlled by a stride parameter. The resulting outputs are therefore filtered pose-direction hypotheses rather than raw grid cells. Figure 5.3 visualises the resulting direction field.

5.5.5 Pose Hypothesis Scoring

Each candidate position-direction pair $(\mathbf{c}_k, \hat{\theta}_k)$ is assigned a score that combines object support and proximity. Let d_{ko} denote the horizontal distance between candidate position $\mathbf{c}_k$ and the floor-projected centroid of object o ,

$$d_{ko} = \|\mathbf{c}_k - \mu_o^{xy}\|_2, \quad (5.30)$$

where $\mu_o^{xy} \in \mathbb{R}^2$ is the projection of the 3D centroid μ_o onto the floor plane.

We define the hypothesis score as

$$s_k = \underbrace{|\mathcal{O}_k^{\text{FoV}}|}_{\text{visibility support}} + \lambda_d \underbrace{\sum_{o \in \mathcal{O}_k^{\text{FoV}}} \exp\left(-\frac{d_{ko}}{\alpha_d}\right)}_{\text{proximity bonus}}, \quad (5.31)$$

where λ_d controls the contribution of the distance term and α_d sets its decay scale. The first term favours poses whose FoV can explain many matched objects simultaneously, while the second term mildly biases the solution toward viewpoints from which these supported objects are also spatially close.

We convert these scores into a posterior distribution over pose-direction hypotheses via a soft-max:

$$P_k = \frac{\exp(s_k/\tau)}{\sum_\ell \exp(s_\ell/\tau)}, \quad (5.32)$$

where τ is a temperature parameter. This posterior provides both a probabilistic localization estimate and, when multi-modal, the uncertainty structure later exploited by the dialog-based disambiguation stage (Section ??).

5.5.6 Density-Based Aggregation of the Final Pose

Selecting the single maximum-scoring hypothesis can be unstable when the posterior contains several nearby peaks caused by repeated object layouts or local symmetries. Instead, we use a density-weighted aggregation strategy that regularises the estimate spatially. This aggregation is well-suited when the posterior is unimodal or contains a dominant cluster of nearby hypotheses, which is the common case in scenes with a single plausible viewpoint region. When the posterior is genuinely multi-modal — e.g. due to symmetric room layouts — the aggregated estimate may fall between modes; this failure case is precisely the scenario addressed by the dialog-based disambiguation stage (Section 5.6), which resolves residual ambiguity through targeted questions rather than spatial averaging.

Let $\mathcal{H}$ denote the full set of scored hypotheses from Section 5.5.5. Although one could threshold this set to retain only high-probability candidates, we include all hypotheses in practice: the Gaussian kernel and the posterior weights P_ℓ jointly ensure that low-probability or spatially isolated hypotheses contribute negligibly to ρ_k , making an explicit pruning step unnecessary. For each $k \in \mathcal{H}$, we compute a local density

$$\rho_k = \sum_{\ell \in \mathcal{H}} P_\ell \exp\left(-\frac{\|\mathbf{c}_k - \mathbf{c}_\ell\|_2^2}{2\sigma^2}\right),$$

where σ is a Gaussian kernel bandwidth. The Gaussian kernel ensures that only hypotheses within a few multiples of σ contribute meaningfully to ρ_k . A hypothesis that is both high-scoring and surrounded by other high-scoring neighbours will receive a large ρ_k , whereas an isolated high-scoring hypothesis — even if individually likely — receives a smaller ρ_k because it lacks local support. The aggregation therefore does not assume that alternatives are nearby; rather, it automatically favours regions of the floor plane where multiple plausible hypotheses concentrate, and down-weights isolated outliers.

The final position estimate is the weighted centroid

$$\hat{\mathbf{c}} = \frac{\sum_{k \in \mathcal{H}} \rho_k \mathbf{c}_k}{\sum_{k \in \mathcal{H}} \rho_k}, \quad (5.33)$$

and the final viewing direction is obtained by aggregating the FoV-consistent direction vectors attached to the same hypotheses. Because directions live on the unit sphere $\mathbb{S}^2$, a simple weighted average $\sum_k \rho_k \hat{\theta}_k / \sum_k \rho_k$ would yield a vector whose norm is strictly less than one whenever the constituent directions are not perfectly aligned. We therefore re-project onto $\mathbb{S}^2$ by normalising by the ℓ_2 -norm of the weighted sum rather than by the sum of weights:

$$\hat{\theta} = \frac{\sum_{k \in \mathcal{H}} \rho_k \hat{\theta}_k}{\left\| \sum_{k \in \mathcal{H}} \rho_k \hat{\theta}_k \right\|_2}. \quad (5.34)$$

This is the standard definition of the mean direction on $\mathbb{S}^2$ and guarantees $\hat{\theta} \in \mathbb{S}^2$ by construction. The final output of the module is therefore $\hat{\mathbf{p}} = (\hat{\mathbf{c}}, \hat{\theta})$. This aggregation step can be interpreted as a spatial regulariser over FoV-aware pose hypotheses. In contrast to a pure MAP rule, it preserves uncertainty structure, suppresses isolated spurious peaks, and yields more stable localization in scenes with repeated or partially symmetric layouts.

5.6 Dialog-based Disambiguation

The fine-localization stage produces a set of pose hypotheses

$$\mathcal{H} = \{h_i = (\mathbf{c}_i, \theta_i)\}_{i=1}^{N_C},$$

where $\mathbf{c}_i \in \mathbb{R}^2$ is a floor-plane position and $\theta_i \in \mathbb{S}^2$ is the associated viewing direction. Their normalised prior probabilities are denoted by $p_C(i)$, with

$$\sum_{i=1}^{N_C} p_C(i) = 1.$$

The dialog stage additionally uses a compact pool of reference frames

$$\mathcal{F} = \{f_j\}_{j=1}^F,$$

selected from the retrieved scene. Each reference frame stores a camera pose together with semantic annotations describing which object labels and spatial relations are visible from that viewpoint. These frame-level annotations provide the semantic grounding used for dialog reasoning.

We consider three Bayesian backends for dialog-based disambiguation that share the same likelihood model and question-selection logic but differ in how the posterior is represented. The main text presents the default backend A3, which maintains a posterior directly over pooled reference frames. The alternative backends A1 and A2 are deferred to Appendix B and treated as ablations.

5.6.1 Reference Frames and Semantic Pooling

After fine localization, we have a set of candidate pose hypotheses

$$\mathcal{H} = \{h_i = (\mathbf{c}_i, \theta_i)\}_{i=1}^{N_C},$$

together with their prior scores from the fine-localization stage. To make dialogue tractable, we do not reason directly over all candidate poses. Instead, we map each candidate hypothesis to a small set of nearby reference frames and construct a pooled subset of informative frames for the selected scene. We denote the resulting pooled reference frames by

$$\mathcal{F}_{\text{pool}} = \{f_j\}_{j=1}^{F_{\text{pool}}}.$$

Each frame f_j stores four quantities that are used by the dialogue backend. First, it stores a 2D floor-plane position $\bar{\mathbf{c}}_j \in \mathbb{R}^2$, obtained by projecting the camera position onto the floor plane. This is the quantity used for pose estimation in the dialogue stage, since the target localization output is a 2D floor position $\hat{\mathbf{c}} \in \mathbb{R}^2$ rather than a full 3D camera centre. Second, it stores a viewing direction $\mathbf{d}_j \in \mathbb{S}^2$. Third, it stores a label-salience dictionary

$$D_j^{\text{lab}} : \ell \mapsto \sigma_j(\ell) \in [0, 1],$$

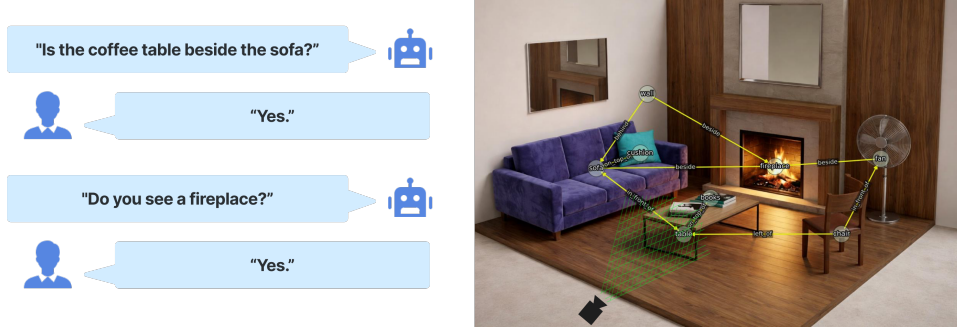


Figure 5.4: DIALOG-BASED DISAMBIGUATION OVERVIEW. Candidate floor poses from the fine-localization grid are mapped via KNN to nearby reference frames, each annotated with visible object labels and spatial relations. At each dialog round, the system selects the yes/no question with maximum information gain (e.g. “Do you see a couch?”), receives the user’s answer, and updates the posterior via Bayes’ rule. The loop repeats until the posterior concentrates above a confidence threshold, at which point the posterior-weighted mean pose is returned.

where $\sigma_j(\ell)$ measures how strongly object label ℓ is visible in frame f_j . Fourth, it stores a set of canonicalized relation triples

$$R_j \subseteq \mathcal{L} \times \mathcal{R} \times \mathcal{L},$$

where each triple (u, r, v) states that object label u stands in relation r to object label v in frame f_j .

The labels used in D_j^{lab} and R_j are canonicalized frame-level labels obtained from the frame metadata by lower-casing and alias normalisation. They are therefore not the same as the unique instance labels introduced earlier for scene-graph retrieval. This normalisation step resolves lexical variants such as synonymous object names and ensures that the same object category is mapped to a consistent label across frames.

The dialogue question pools are then built from the pooled frames. The label question pool contains all canonicalized labels that appear in $\mathcal{F}_{\text{pool}}$, while the relation question pool contains all canonicalized relation triples that appear in $\mathcal{F}_{\text{pool}}$. Both pools are therefore scene-specific and depend on the pooled reference frames rather than on a global dataset-wide dictionary.

5.6.2 Frame-Level Posterior (Backend A3)

The default dialogue backend used in the main pipeline is A3, which maintains a posterior over pooled reference frames. The alternative backends A1 and A2 are described in Appendix B and treated as ablations.

Let $p_0(h_i)$ denote the prior probability of candidate pose hypothesis h_i from the fine-localization stage. We map each candidate hypothesis to the pooled reference frames using a row-normalised candidate-to-frame weight matrix

$$W \in \mathbb{R}^{N_C \times F_{\text{pool}}}, \quad W_{ij} \geq 0, \quad \sum_{j=1}^{F_{\text{pool}}} W_{ij} = 1.$$

The entry W_{ij} measures how strongly candidate hypothesis h_i is associated with pooled frame f_j .

The initial frame prior is induced from the fine-localization posterior by marginalizing candidate hypotheses onto pooled frames:

$$p_0(j) = \frac{\sum_{i=1}^{N_C} p_0(h_i) W_{ij}}{\sum_{j'=1}^{F_{\text{pool}}} \sum_{i=1}^{N_C} p_0(h_i) W_{ij'}}. \quad (5.35)$$

Thus, the dialogue stage does not start from a uniform prior. Instead, it inherits the uncertainty structure of the fine-localization stage and refines it at frame level.

At dialog round r , backend A3 maintains a posterior

$$p_r(j) = P(J = j),$$

where $J \in \{1, \dots, F_{\text{pool}}\}$ is the latent pooled-frame hypothesis. This discrete posterior is the belief state updated throughout the dialogue. It is also the distribution used for question selection and for computing the final dialog-based pose estimate.

5.6.3 Likelihood Model

Let q denote the current dialog question and let $a \in \{\text{yes}, \text{no}, \text{unknown}\}$ denote the observed answer. The likelihood term

$$\mathcal{L}(a | j, q)$$

therefore means the probability of receiving answer a if pooled frame f_j were the true state and question q were asked.

We consider two question types. A label question has the form $q = \ell$ and asks whether object label ℓ is visible. A relation question has the form $q = (u, r, v)$ and asks whether relation r holds between labels u and v .

For a label question $q = \ell$, we derive two soft quantities from the label salience score $\sigma_j(\ell)$. The first is a soft truth probability

$$p_{\text{true}}(j, q) = \min\left(\frac{\sigma_j(\ell)}{\tau_{\text{vis}}}, 1\right),$$

which models how likely it is that the correct answer is “yes.” The second is an answerability probability

$$p_{\text{ans}}(j, q) = \min\left(\frac{\sigma_j(\ell)}{\tau_{\text{ans}}}, 1\right),$$

which models how confidently the question can be answered from frame f_j .

For a relation question $q = (u, r, v)$, the truth probability is binary at frame level:

$$p_{\text{true}}(j, q) = \begin{cases} 1, & \text{if } (u, r, v) \in R_j, \\ 0, & \text{otherwise.} \end{cases}$$

The corresponding answerability is defined from the visibility of the endpoint labels:

$$p_{\text{ans}}(j, q) = \min(p_{\text{ans}}(j, u), p_{\text{ans}}(j, v)).$$

To model uncertain or unanswerable questions, we define the probability of the answer “unknown” as

$$p_u(j, q) = p_u^{\text{base}} p_{\text{ans}}(j, q) + p_u^{\text{unans}} (1 - p_{\text{ans}}(j, q)), \quad (5.36)$$

where p_u^{base} and p_u^{inans} are likelihood hyperparameters. The first controls the baseline probability of an “unknown” response even for answerable questions, while the second increases that probability when the question is not answerable from frame f_j .

A further likelihood hyperparameter is the reliability term $\alpha_q \in (0, 1)$, which controls how strongly the model trusts the underlying proposition queried by q . Intuitively, higher values make the update more confident in the corresponding yes/no evidence, while lower values produce more conservative updates. We use separate reliability values for label and relation questions, with a lower value for relation questions because relation cues are noisier than binary object-presence cues. The precise values used in evaluation are listed in Appendix section A.5.

With this notation, the full three-way likelihood is

$$\mathcal{L}(\text{yes} \mid j, q) = (1 - p_u(j, q)) \left[\alpha_q p_{\text{true}}(j, q) + (1 - \alpha_q)(1 - p_{\text{true}}(j, q)) \right], \quad (5.37)$$

$$\mathcal{L}(\text{no} \mid j, q) = (1 - p_u(j, q)) \left[(1 - \alpha_q) p_{\text{true}}(j, q) + \alpha_q (1 - p_{\text{true}}(j, q)) \right], \quad (5.38)$$

$$\mathcal{L}(\text{unknown} \mid j, q) = p_u(j, q). \quad (5.39)$$

This likelihood explicitly models both soft truth and answerability, and it treats the answer “unknown” as informative rather than discarding it.

5.6.4 Bayesian Update

At dialog round r , the system selects a question q_r and receives an answer $a_r \in \{\text{yes}, \text{no}, \text{unknown}\}$. In the actual evaluation protocol, these answers are obtained from a human annotator for 3RScan and from Qwen for ScanNet, where Qwen answers using the ground-truth reference-frame metadata. Given the current posterior $p_{r-1}(j)$ and the likelihood model from Section 5.6.3, the posterior is updated by Bayes’ rule:

$$p_r(j) = \frac{p_{r-1}(j) \mathcal{L}(a_r \mid j, q_r)}{\sum_{j'=1}^{F_{\text{pool}}} p_{r-1}(j') \mathcal{L}(a_r \mid j', q_r)}. \quad (5.40)$$

This update has the expected behaviour. Frames that are more compatible with the observed answer gain posterior mass, while incompatible frames are suppressed. Because the likelihood includes a non-trivial “unknown” branch, an answer of “unknown” also changes the belief state in a principled way rather than being ignored. Repeating this update over dialog rounds progressively concentrates the posterior on a smaller set of plausible pooled frames.

5.6.5 Question Selection

Let J denote the latent pooled-frame hypothesis and let $A \in \{\text{yes}, \text{no}, \text{unknown}\}$ denote the answer random variable for a candidate dialog question q . At each round, we greedily select the question that maximises expected information gain under the current posterior p_r . The score assigned to a candidate question is

$$s(q) = H(p_r) - \sum_{a \in \{\text{yes}, \text{no}, \text{unknown}\}} P(A = a \mid q) H(P(J \mid A = a, q)), \quad (5.41)$$

where the entropy of the current posterior is

$$H(p_r) = - \sum_j p_r(j) \log p_r(j), \quad (5.42)$$

and the marginal probability of receiving answer a to question q is

$$P(A = a | q) = \sum_j p_r(j) \mathcal{L}(a | j, q). \quad (5.43)$$

This score therefore measures the current entropy minus the expected posterior entropy after asking question q .

In practice, we apply several lightweight heuristics to improve the usefulness of the selected question. Very common labels such as `floor`, `wall`, and `ceiling` are excluded entirely, since they are rarely discriminative. For every remaining label question, the score is reweighted by an inverse-document-frequency term computed over the pooled reference frames:

$$s'_{\text{lab}}(q) = s(q) (1 + \lambda_{\text{idf}} \text{IDF}(\ell)), \quad \text{IDF}(\ell) = \log \frac{F_{\text{pool}} + 1}{df(\ell) + 1}, \quad (5.44)$$

where ℓ denotes the queried label and $df(\ell)$ denotes the number of pooled frames in which ℓ appears. The pooled frames therefore play the role of documents in the IDF computation. Labels that occur in many pooled frames receive little or no additional boost, whereas scene-specific labels receive a larger boost.

Relation questions receive a small multiplicative bonus because they are typically more discriminative than single-object visibility questions. We also exclude questions whose answer is already near-certain under the current posterior. Concretely, questions are filtered out when

$$P(A = \text{yes} | q) \notin [0.01, 0.99]. \quad (5.45)$$

For relation questions, we additionally require the expected answerability

$$\sum_j p_r(j) p_{\text{ans}}(j, q) \quad (5.46)$$

to exceed a minimum threshold before the question is asked. All of these rules affect only which question is selected next; they do not modify the Bayesian update itself.

5.6.6 Stopping Criteria and Final Pose

The dialogue terminates when one of three conditions is met. First, after a minimum number of rounds, the system stops if the frame posterior is sufficiently concentrated, i.e. if

$$\max_j p_r(j) \geq \tau_{\text{conf}}.$$

Second, the dialogue stops if the maximum number of rounds is reached. Third, it also stops if no admissible question remains after applying the selection rules from Section 5.6.5.

Let R denote the final dialog round and let

$$j^* = \arg \max_j p_R(j)$$

be the maximum-a-posteriori pooled frame. Backend A3 outputs two final pose estimates, both of which are reported in evaluation. The MAP estimate is

$$(\hat{\mathbf{c}}_{\text{MAP}}, \hat{\theta}_{\text{MAP}}) = (\bar{\mathbf{c}}_{j^*}, \mathbf{d}_{j^*}), \quad (5.47)$$

which corresponds to the single most probable pooled frame. In addition, we compute a posterior-weighted mean estimate:

$$\hat{\mathbf{c}}_{\text{mean}} = \sum_j p_R(j) \bar{\mathbf{c}}_j, \quad \hat{\boldsymbol{\theta}}_{\text{mean}} = \frac{\sum_j p_R(j) \mathbf{d}_j}{\left\| \sum_j p_R(j) \mathbf{d}_j \right\|_2}. \quad (5.48)$$

These two outputs serve complementary roles. The MAP estimate returns the single most likely pooled frame, while the mean estimate summarises the full posterior distribution. Since the localization target is a 2D floor-plane pose, the final position estimate uses the floor-plane coordinates $\bar{\mathbf{c}}_j \in \mathbb{R}^2$, whereas the viewing direction remains a 3D unit vector $\mathbf{d}_j \in \mathbb{S}^2$.

Results

6.1 Coarse Localization

6.1.1 Evaluation protocol

Overview. We evaluate scene retrieval under three protocols that probe different axes of generalisation: candidate-set difficulty, and query-distribution shift. Each protocol specifies (i) the type of text query, (ii) the *candidate set* against which each query is ranked, and (iii) the rank cutoffs k at which Recall is reported. This evaluation protocol is designed to directly benchmark against the retrieval system of [Chen et al. \(2024\)](#), following their protocol definitions so that results are directly comparable.

A *candidate set* $\mathcal{C}_q$ is the finite collection of 3D scene graphs that the retrieval system must rank for a given query q . Rather than ranking every scene in the database, we follow standard practice and form $\mathcal{C}_q$ from the ground-truth scene together with a number of *negatives*—scenes that are not the correct match and therefore act as distractors. Negatives are required because a retrieval system is only useful if it can distinguish the correct scene from plausible alternatives; a model that assigns high scores to all scenes indiscriminately would still rank the ground truth first by chance in a trivially small candidate set. Controlling the size and source of the negative pool therefore directly controls the difficulty of the evaluation.

Recall@ k . All retrieval results are reported as Recall@ k (R@ k): the fraction of queries for which the correct scene appears within the top- k ranked candidates. Formally, given a query T with ground-truth scene S^* , the retrieval system produces a ranked list $\pi(T) = (i_1, i_2, \dots, i_{|\mathcal{C}_q|})$ by sorting all candidates in descending order of the scoring function (5.25). The binary relevance indicator at rank cutoff k is:

$$\text{rel}_k(T) = \mathbb{I}[S^* \in \{S_{i_1}, \dots, S_{i_k}\}] \in \{0, 1\}, \quad (6.1)$$

and Recall@ k over a set of Q queries is:

$$\text{R@}k = \frac{1}{Q} \sum_{q=1}^Q \text{rel}_k(T_q) \in [0, 1]. \quad (6.2)$$

Scoring weights via grid search. The scoring function (5.25) combines three components—embedding similarity, global graph similarity, and Jaccard overlap—with scalar weights w_{emb} , w_{glob} , and w_{jac} . We select these weights via *grid search*: a systematic sweep over a discrete set of candidate values for each weight, evaluating every combination on a held-out validation set and retaining

the combination that maximises mean Recall. Grid search is appropriate here because the scoring function is non-differentiable with respect to its discrete graph-matching components, ruling out gradient-based hyperparameter optimisation. We found that $w_{\text{emb}} = 0.33$, $w_{\text{glob}} = 0.33$, $w_{\text{jac}} = 0.34$ achieved the best validation performance.

Variance reduction. Because negatives are sampled randomly, a single evaluation round can exhibit high variance. To obtain stable estimates, each protocol is repeated for $N = 10$ independent rounds of $Q = 100$ queries each, with a fresh negative sample drawn every round. The reported mean and standard deviation are:

$$\overline{\text{R@}k} = \frac{1}{N} \sum_{n=1}^N \text{R@}k^{(n)}, \quad (6.3)$$

$$\sigma_{\text{R@}k} = \sqrt{\frac{1}{N} \sum_{n=1}^N (\text{R@}k^{(n)} - \overline{\text{R@}k})^2}, \quad (6.4)$$

where $\text{R@}k^{(n)}$ is the recall in round n . All results are reported as $\overline{\text{R@}k} \times 100 \pm \sigma_{\text{R@}k} \times 100$ (percentage).

Three protocols. We evaluate under three protocols following [Chen et al. \(2024\)](#); baselines are taken directly from that work. The protocols are designed to test complementary aspects of the retrieval system.

Protocol 1 and **Protocol 2** both use *ScanScribe text graphs* as queries—structured graphs derived from manual scene-graph annotations. Because ScanScribe annotations exist for the same scenes that appear in the test set, the training pool for these two protocols contains scenes that *overlap* with the test set; however, the text-graph queries used at test time are distinct subgraphs of those scenes and were not seen during training. This setting therefore tests in-distribution generalisation: can the model retrieve a scene when it has been exposed to other views of that scene during training. Protocol 1 uses a small candidate set of 10 scenes to measure precision in a low-distractor regime, while Protocol 2 uses the full test set as the candidate pool, making it strictly harder.

Protocol 3 replaces the structured ScanScribe queries with text graphs parsed from free-form GPT-4o-mini descriptions of scene images. More importantly, the training data for Protocol 3 contains *no scenes that overlap with the test set*, so the model must generalise to both a new query distribution and previously unseen scenes simultaneously. This is the most challenging protocol and the primary indicator of out-of-distribution generalisation. Because the 3D scene graph database and the ranking function are identical across all three protocols, any performance gap between Protocol 1 and Protocol 3 is attributable entirely to the query distribution shift and the absence of overlapping training scenes.

Formally, the candidate set for each query is:

$$\mathcal{C}_q = \begin{cases} \{S^*\} \cup \mathcal{U}(\mathcal{D}_{\text{pool}} \setminus \{S^*\}, 9) & \text{Protocols 1 and 3, } k \in \{1, 2, 3, 5\}, \\ \mathcal{D}_{\text{test}} & \text{Protocol 2, } k \in \{5, 10, 20, 30\}, \end{cases} \quad (6.5)$$

where $\mathcal{U}(\mathcal{D}_{\text{pool}} \setminus \{S^*\}, 9)$ denotes a uniform random sample of 9 scenes drawn *without replacement* from the pool $\mathcal{D}_{\text{pool}}$ after removing the ground-truth scene, $|\mathcal{D}_{\text{pool}}| = 218$, and $|\mathcal{D}_{\text{test}}| = 55$.

Baselines. We compare against three baselines taken directly from [Chen et al. \(2024\)](#), which we do not re-run. A key distinction between the baseline methods and our pipeline concerns both the embedding strategy and the training regime.

Text2Pos [Kolmet et al. \(2022\)](#) learns a joint embedding between free-form text and outdoor point clouds. It requires a fixed known-words vocabulary mapping, which severely limits its ability to generalise to open-set indoor text queries. It is included as a lower bound.

CLIP2CLIP matches scenes by computing cosine similarity between CLIP embeddings of the text query and rendered scene views, with no structural reasoning over object relations. It requires no training on the ScanScribe data and generalises reasonably to out-of-distribution queries precisely because it relies on CLIP’s large-scale pretraining rather than task-specific supervision.

Text2SGM [Chen et al. \(2024\)](#) is the closest prior work and the primary point of comparison. It trains a joint graph embedding model using *Word2Vec* [Mikolov et al. \(2013\)](#) 300-dimensional embeddings for both node labels and edge relation strings, with node features augmented by object attributes. The model is trained contrastively on *text graph – 3D scene graph pairs* drawn from the ScanScribe dataset — that is, it is directly supervised on matched text and scene graph pairs from the same scenes that appear in the test set. Three scoring variants are evaluated: match-prob (a learned binary matching probability), cos-sim (cosine similarity of joint embeddings), and ret-based (a retrieval-based scoring mechanism).

Our pipeline differs from Text2SGM in two fundamental ways. First, we replace Word2Vec with *CLIP* embeddings for all node labels and relation strings, operating in a continuous 512-dimensional semantic space rather than a fixed vocabulary lookup. This allows synonymous relation strings from different sources to be naturally aligned without explicit vocabulary normalisation. Second, our model is trained on *pairs of 3D scene graphs* — never on text graphs — using subgraph augmentation to simulate the partial observations characteristic of natural language queries. Under Protocols 1 and 2, both Text2SGM and our method train and evaluate on scenes from the same pool ($|\mathcal{D}_{\text{pool}}| = 218$), so both benefit from in-distribution scene exposure. Under Protocol 3, however, the LLM-generated queries are derived from scenes with *no overlap* with the training pool, making it a true out-of-distribution test. Text2SGM’s performance collapses by -34.4 percentage points under this shift, while ours degrades by only -0.6 points — a direct consequence of CLIP’s vocabulary-agnostic semantic alignment versus Word2Vec’s dependency on the training annotation vocabulary.

Table 6.1: SCENE RETRIEVAL RECALL@ k ON SCANSCRIBE (10-SCENE POOL). *Scene retrieval Recall@ k (%) on ScanScribe ([Chen et al., 2024](#)). Each query is matched against a pool of 10 candidate scenes. Best in **bold**.*

Method	Top-1	Top-2	Top-3	Top-5
Text2Pos (scratch)	11.63 $\pm$ 0.99	22.57 $\pm$ 1.16	33.03 $\pm$ 1.36	52.80 $\pm$ 1.67
Text2Pos (fine-tuning)	10.30 $\pm$ 0.93	20.74 $\pm$ 1.32	31.65 $\pm$ 1.45	53.42 $\pm$ 1.68
CLIP2CLIP	33.09 $\pm$ 0.21	52.53 $\pm$ 0.24	65.98 $\pm$ 0.21	82.69 $\pm$ 0.13
Text2SGM (match-prob)	63.70 $\pm$ 1.85	81.40 $\pm$ 1.85	89.31 $\pm$ 1.42	95.71 $\pm$ 0.93
Text2SGM (cos-sim)	68.27 $\pm$ 2.05	84.95 $\pm$ 1.62	91.65 $\pm$ 1.23	97.14 $\pm$ 0.76
Text2SGM (ret-based)	68.61 $\pm$ 0.04	85.00 $\pm$ 0.02	91.57 $\pm$ 0.01	96.99 $\pm$ 0.00
LangLoc (ours)	76.70 $\pm$ 4.58	90.40 $\pm$ 2.73	96.10 $\pm$ 1.58	98.90 $\pm$ 1.22

Full 55-scene test set (Table 6.2). Retrieval over all 55 test scenes is a substantially harder task, as the candidate pool is $5.5\times$ larger than the 10-scene setting. LangLoc reaches 83.30% Top-

5 and 91.60% Top-10, outperforming the strongest Text2SGM variant (cos-sim) by +7.0 and +3.8 percentage points respectively. Text2Pos drops to around 10% at Top-5, while CLIP2CLIP recovers marginally to 33.26% Top-5, still far below all graph-based methods. The consistent margin over Text2SGM across all k values confirms that the dual-branch encoder with gated fusion learns more discriminative scene embeddings than single-branch alternatives.

Table 6.2: SCENE RETRIEVAL RECALL@ k ON THE FULL 55-SCENE TEST SET. *Scene retrieval Recall@ k (%) on ScanScribe (Chen et al., 2024), retrieving from the full set of 55 test scenes. Best in bold.*

Method	Top-5	Top-10	Top-20	Top-30
Text2Pos (scratch)	10.27 $\pm$ 0.83	21.24 $\pm$ 1.17	39.36 $\pm$ 1.43	57.30 $\pm$ 1.48
Text2Pos (fine-tuning)	9.21 $\pm$ 0.92	18.27 $\pm$ 1.24	39.27 $\pm$ 1.61	58.63 $\pm$ 1.60
CLIP2CLIP	33.26 $\pm$ 0.21	53.47 $\pm$ 0.23	73.76 $\pm$ 0.18	86.23 $\pm$ 0.11
Text2SGM (match-prob)	70.24 $\pm$ 1.93	85.71 $\pm$ 1.38	93.36 $\pm$ 1.06	97.25 $\pm$ 0.65
Text2SGM (cos-sim)	76.34 $\pm$ 2.06	87.83 $\pm$ 1.55	95.40 $\pm$ 0.89	98.24 $\pm$ 0.57
Text2SGM (ret-based)	76.29 $\pm$ 0.16	87.77 $\pm$ 0.10	95.34 $\pm$ 0.05	98.18 $\pm$ 0.02
LangLoc (ours)	83.30 $\pm$ 3.74	91.60 $\pm$ 3.01	97.10 $\pm$ 1.51	98.80 $\pm$ 0.87

LLM-generated queries (Table 6.3). To evaluate robustness under domain shift, queries are generated by GPT-4o-mini from scene images rather than derived from scene graph annotations, reducing lexical overlap between query terms and database labels. This is a significantly harder setting for Text2SGM, whose performance collapses from 68.6% to 34.2% Top-1 (a drop of -34.4 percentage points), revealing a strong dependence on lexical overlap between text graph labels and scene graph labels. CLIP2CLIP is unaffected (33.1% Top-1), as it does not rely on label matching. LangLoc obtains 76.1% Top-1, a +42.0 percentage point gain over Text2SGM and +43.0 over CLIP2CLIP, and reaches 98.6% at Top-5. The near-zero degradation from the standard protocol (76.7% $\rightarrow$ 76.1% Top-1) demonstrates that LangLoc’s retrieval performance is robust to phrasing variability and lexical mismatch.

Table 6.3: SCENE RETRIEVAL RECALL@ k WITH LLM-GENERATED QUERIES. *Scene retrieval Recall@ k (%) on ScanScribe (Chen et al., 2024) using LLM-generated queries (GPT-4o-mini from scene images). Best in bold.*

Method	Top-1	Top-2	Top-3	Top-5
CLIP2CLIP	33.07 $\pm$ 0.21	52.52 $\pm$ 0.23	66.00 $\pm$ 0.21	82.78 $\pm$ 0.13
Text2SGM (cos-sim)	34.22 $\pm$ 1.77	56.67 $\pm$ 1.70	68.78 $\pm$ 2.35	82.11 $\pm$ 1.23
LangLoc (ours)	76.10 $\pm$ 3.48	88.50 $\pm$ 3.14	94.20 $\pm$ 2.27	98.60 $\pm$ 1.02

6.2 Fine Localization

Given the retrieved scene, we evaluate how accurately LangLoc estimates the observer’s 2D floor position $\hat{c}$ and heading $\hat{\theta}$.

6.2.1 Evaluation Protocol

Metrics. We report four metrics:

- **Position error:** Euclidean distance $e_{\text{pos}} = \|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$ in metres.
- **Top- K position error:** Minimum distance among the $K=10$ highest-scoring grid cells,

$$e_{\text{top-}K} = \min_{k \in \text{top-}K} \|\mathbf{c}_k - \mathbf{c}^*\|_2.$$

This softer metric captures whether the correct location appears as a plausible mode in the posterior, even if it is not the single highest-scoring cell.

- **Angular error:** The angle between predicted and ground-truth direction vectors, reported in degrees.
- **3D View IoU:** Frustum intersection-over-union between predicted and ground-truth viewing frustums, computed as an area-weighted overlap of visible mesh triangles. This jointly evaluates position and orientation accuracy.

Baselines. We compare against two baselines that require no task-specific training:

- **Midpoint:** Places the observer at the centroid of the floor bounding box. It uses no language input and does not predict a heading.
- **VLM (Qwen2.5-VL):** Provides Qwen2.5-VL-2B (Wang et al., 2024) with a top-down rendering of the scene and the text query, and asks it to predict pixel coordinates and a heading in a zero-shot setting. Predicted pixel locations are unprojected to 3D via known intrinsics and floor-plane intersection.

6.2.2 Results on 100-Scene Subsets

Table 6.4 and Table 6.5 present results on 100-scene subsets of 3RScan and ScanNet respectively. On both datasets, LangLoc without dialog already halves the angular error compared to the VLM baseline, demonstrating that ray-cast visibility scoring yields meaningful heading estimates from language alone. The VLM produces near-random headings ($\sim 85\text{--}94^\circ$) and near-zero IoU despite predicting a heading.

Adding dialog dramatically improves all metrics. On 3RScan, the median position error drops from 1.55 m to 0.80 m (a 48% improvement) and the IoU nearly doubles from 0.172 to 0.342. On ScanNet, the gains are even larger: the median position error drops from 1.31 m to 0.07 m, the median angular error from 34.7° to 5.1° , and the IoU rises from 0.236 to 0.593.

An important pattern emerges in the raw position error: the midpoint baseline achieves seemingly competitive numbers despite using *no* language at all. This is an artefact of small indoor rooms: the floor centroid is mechanically close to every point in a compact space. The Top-10 metric disambiguates this: the correct position is almost always among the high-scoring cells, confirming that LangLoc identifies the right region even when the single best cell is not exact.

6.2.3 Full-Scale Evaluation

Table 6.6 reports results on the complete LangLoc dataset (1,300 scenes, 13k+ descriptions) without dialog. LangLoc achieves a Top-10 median position error of 0.95 m and a median angular error of 39.8° , consistent with the 100-scene subset and confirming that the method scales to the full benchmark. LangLoc’s 3D IoU of 0.147 is an order of magnitude higher than the VLM’s 0.018, confirming that the method recovers both accurate position and orientation from language alone.

Table 6.4: FINE LOCALIZATION ON 3RSCAN. *Fine localization on 3RScan (100-scene subset). Best in **bold**, second best underlined.*

Method	Top-10 Pos. (m) ↓		Pos. (m) ↓		Angle (deg) ↓		3D IoU ↑
	Mean	Med.	Mean	Med.	Mean	Med.	Mean
Midpoint	–	–	<u>1.416</u>	<u>1.347</u>	–	–	–
VLM (Qwen2.5-VL)	–	–	1.582	1.420	85.48	85.54	0.062
LangLoc w/o dialog	<u>1.037</u>	<u>0.941</u>	1.712	1.551	<u>46.07</u>	<u>37.24</u>	<u>0.172</u>
LangLoc w/ dialog	0.983	0.890	0.926	0.799	39.52	33.37	0.342

Table 6.5: FINE LOCALIZATION ON SCANNET. *Fine localization on ScanNet (100-scene subset). Best in **bold**, second best underlined.*

Method	Top-10 Pos. (m) ↓		Pos. (m) ↓		Angle (deg) ↓		3D IoU ↑
	Mean	Med.	Mean	Med.	Mean	Med.	Mean
Midpoint	–	–	<u>1.279</u>	<u>1.098</u>	–	–	–
VLM (Qwen2.5-VL)	–	–	1.382	1.099	93.96	91.27	0.030
LangLoc w/o dialog	<u>1.254</u>	<u>1.065</u>	1.676	1.314	<u>42.67</u>	<u>34.66</u>	<u>0.236</u>
LangLoc w/ dialog	0.532	0.210	0.593	0.069	23.77	5.12	0.593

Table 6.6: FINE LOCALIZATION ON THE FULL LANGLOC DATASET. *Fine localization on the full LangLoc dataset (1,300 scenes). Best in **bold**, second best underlined.*

Method	Top-10 Pos. (m) ↓		Pos. (m) ↓		Angle (deg) ↓		3D IoU ↑
	Mean	Med.	Mean	Med.	Mean	Med.	Mean
Midpoint	–	–	1.369	1.259	–	–	–
VLM (Qwen2.5-VL)	–	–	1.538	1.350	<u>84.70</u>	<u>81.70</u>	<u>0.018</u>
LangLoc w/o dialog	1.153	0.951	<u>1.534</u>	<u>1.308</u>	46.85	39.80	0.147

6.2.4 Qualitative Results

Figure 6.1 shows qualitative frustum overlap visualisations on 3RScan scenes. The ground-truth frustum (red) and predicted frustum (yellow) overlap substantially (orange) when the method correctly identifies both position and heading.



Figure 6.1: QUALITATIVE FRUSTUM OVERLAP ON 3RSCAN. Ground-truth frustum (red), predicted frustum (yellow), and their overlap (orange). The method correctly identifies both position and heading in diverse room configurations.

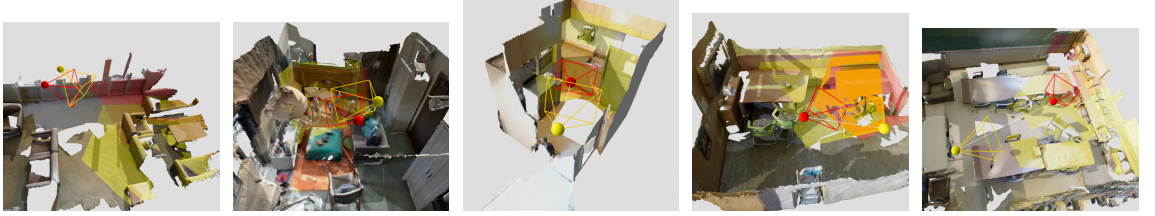


Figure 6.2: QUALITATIVE FRUSTUM OVERLAP ON SCANNET. Ground-truth frustum (red), predicted frustum (yellow), and their overlap (orange) on five ScanNet scenes. The method generalises across datasets without retraining.

6.3 Dialog-based Disambiguation

To evaluate dialog-based disambiguation, we initialise the dialog from the fine-localization posterior and run it for at most 12 rounds. For 3RScan, answers are provided interactively by a human annotator. For ScanNet, answers are generated automatically by the Qwen-based protocol described in Appendix A.6, which uses ground-truth reference-frame metadata. Unless stated otherwise, we report the default A3 backend in the main text; the full comparison with A1 and A2 is given in Appendix B.

Table 6.7 shows one example dialog transcript for `scene0002_00`. The system begins with object-visibility questions and introduces relation questions only when further disambiguation is needed. Each answer is incorporated by Bayes’ rule, progressively concentrating posterior mass on the correct region of the scene.

Table 6.7: EXAMPLE DIALOG TRANSCRIPT FOR `scene0002_00`. Questions are selected by expected information gain. For this ScanNet example, answers are produced by Qwen from the ground-truth reference-frame metadata.

Round	Question	Answer	$\max p$ (A3)
Init	(initial)	—	0.17
1	Do you see a couch?	Yes	0.42
2	Do you see a door?	Unk	0.49
3	Do you see a window?	Yes	0.86

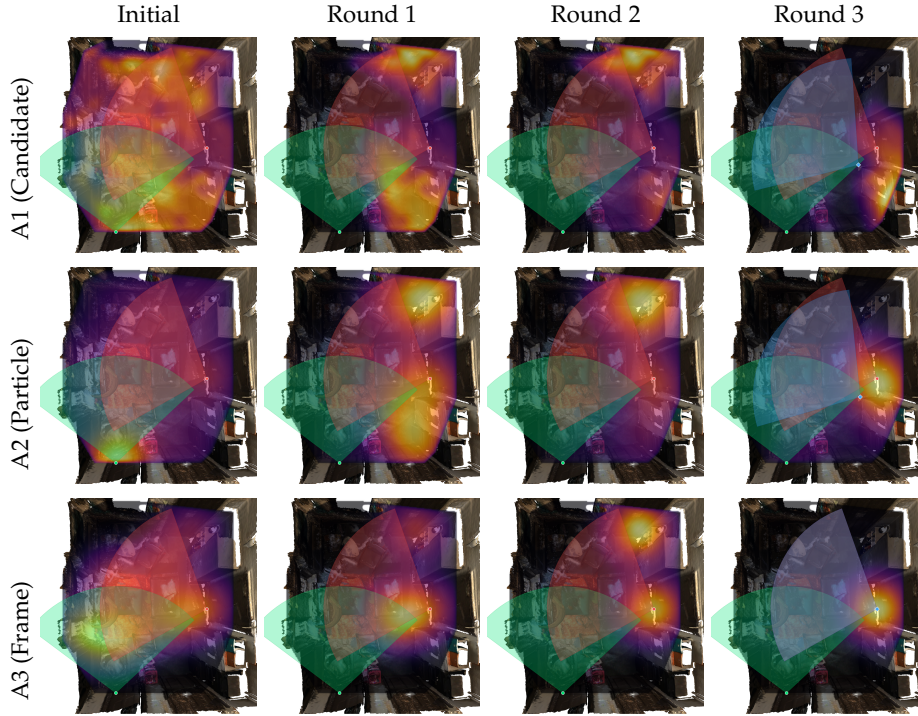


Figure 6.3: DIALOG TRACE FOR `scene0002_00`. Posterior heatmaps for all three backends are shown across successive dialog rounds. The ground-truth pose is shown in red, the initial fine-localization prediction in green, and the final dialog-refined pose in blue. A3 concentrates most rapidly because it reasons directly over pooled reference frames with exact frame-level semantics.

Figure 6.3 provides a qualitative view of how the posterior changes over successive dialog rounds for the three backend variants on `scene0002_00`. In this example, the posterior under A3 becomes visibly more concentrated around the correct region after three questions, with the maximum posterior probability increasing from 0.17 to 0.86. The A1 and A2 traces also become more focused over time, but remain more diffuse in this particular example. This qualitative behaviour is consistent with the quantitative backend comparison in Appendix B, which evaluates all three variants systematically across the dataset.

The quantitative impact of dialog is substantial. On 3RScan, the median position error decreases from 1.55 m without dialog to 0.80 m with dialog, corresponding to a 48% improvement, while the mean 3D IoU increases from 0.172 to 0.342. On ScanNet, the gains are larger still: the median position error decreases from 1.31 m to 0.07 m, and the mean 3D IoU increases from 0.236 to 0.593. These results show that a small number of targeted questions is often sufficient to resolve the residual spatial ambiguity left by a single free-form description.

Discussion

7.1 Scene Retrieval

7.1.1 Comparison with Text2SGM

Text2SGM [Chen et al. \(2024\)](#) is the most relevant prior work, as it addresses the same task, matching text descriptions against 3D scene graphs, using a similar graph-matching framework. The consistent margin of LangLoc over all three Text2SGM variants across all evaluation protocols warrants careful analysis of what drives the improvement.

Standard retrieval protocols. Under the 10-scene pool (Table 6.1) and full 55-scene test set (Table 6.2), LangLoc outperforms the strongest Text2SGM variant by +8.1 and +7.0 percentage points at Top-1 and Top-5 respectively. The gains are consistent across all values of k , suggesting that the dual-branch encoder with gated fusion learns genuinely more discriminative scene embeddings rather than simply reranking the top candidates differently.

Robustness to unseen text graphs. The most revealing comparison occurs under the LLM-generated query protocol (Table 6.3), where queries are generated freely from scene images by GPT-4o-mini and constitute genuinely unseen text graphs, i.e. graphs whose vocabulary, phrasing, and structure were never observed during training by either method. Text2SGM collapses from 68.6% to 34.2% Top-1 under this shift, a degradation of 34.4 percentage points. LangLoc degrades by only −0.6 percentage points (76.7% → 76.1% Top-1), demonstrating near-complete robustness to unseen text graphs.

This asymmetry reveals a structural weakness in Text2SGM: its embeddings are tightly coupled to the annotation vocabulary of the ScanScribe training set, and degrade sharply when query phrasing no longer mirrors that vocabulary. LangLoc avoids this by embedding all node labels and relation strings with CLIP, which provides soft semantic alignment in a continuous space where synonymous expressions are naturally close regardless of surface form. This robustness to unseen text graphs is an emergent property of the CLIP-based encoding and represents a practically important advantage: in real deployment, a user’s natural language description will inevitably use phrasing that was never seen during training, and a system that degrades by only −0.6 percentage points under such shift is substantially more reliable than one that loses 34.4 points.

7.1.2 Role of Unique Instance Labelling

Removing unique instance labels causes complete training collapse: the InfoNCE loss stagnates and no discriminative embeddings are learned. This outcome can be understood directly from the contrastive objective: when all instances of the same class share an identical CLIP embedding $\phi(l_i^{\text{base}})$, the graph embeddings $\mathbf{z}_i$ and $\mathbf{z}_j$ become indistinguishable across scenes, the softmax in (2.11) outputs a near-uniform distribution, and the gradient vanishes.

With unique labels, two wall instances from different spatial zones — `wall_north_upper` and `wall_south_lower` — satisfy

$$\cos(\phi(l_i^{\text{unique}}), \phi(l_j^{\text{unique}})) < \cos(\phi(l_i^{\text{base}}), \phi(l_j^{\text{base}})) = 1, \quad (7.1)$$

restoring the input entropy required for the contrastive loss to provide a useful gradient signal. The performance gap is therefore not a matter of degree but of kind: unique labelling is a necessary precondition for training, not an optional regulariser.

7.1.3 Role of Subgraph Augmentation

Subgraph augmentation is motivated by a fundamental mismatch between the training and inference distributions. At inference, a sparse text graph G^t — containing only the objects explicitly mentioned in a description — must be matched against a dense full 3D scene graph G^s containing all annotated instances. Without any augmentation, the encoder is trained exclusively on dense graph pairs and may learn embeddings that exploit whole-graph statistics such as node count or global density, which are systematically different between a sparse text query and a full scene graph.

By randomly retaining 40–70% of nodes during training, subgraph augmentation prevents the encoder from relying on these whole-graph statistics and forces it to identify a scene from a partial set of object mentions. However, we acknowledge the limitation raised by this design: subgraph-to-subgraph training does not automatically guarantee generalisation to subgraph-to-full-graph matching at inference. The two settings differ not only in the density of the query graph but also in the asymmetry between query and database: at training time both graphs are partial, whereas at inference the database graph is always the full scene. A controlled ablation isolating this effect would be needed to confirm the benefit empirically, and we leave this as a direction for future work. Nevertheless, we consider subgraph augmentation a reasonable inductive bias given that the alternative — training on full graphs only — would expose the encoder to a distribution even further from the sparse text queries it must handle at inference.

7.1.4 Limitations of Scene Retrieval

Despite strong retrieval performance, several limitations remain. The model operates on pre-built 3D scene graphs with fixed object vocabularies; it cannot handle open-vocabulary descriptions that mention objects outside the 3DSSG label set. Performance degrades in scenes with many repeated object types even with unique labelling, since the spatial qualifiers (`north`, `upper`, etc.) may not fully disambiguate a large number of identical instances. The inference weights $w_{\text{emb}} = 0.33$, $w_{\text{glob}} = 0.33$, $w_{\text{jac}} = 0.34$ were set to near-uniform values on the validation split; a more exhaustive weight search may yield further gains. Finally, the variance on LangLoc’s results (e.g. ± 4.58 at Top-1 in Table 6.1) is notably higher than Text2SGM’s near-zero variance, suggesting sensitivity to random negative sampling that could be reduced by averaging over more evaluation runs.

7.2 Fine Localization

7.2.1 The Midpoint Baseline Artefact

A counter-intuitive finding is that the midpoint baseline, which uses no language input whatsoever, achieves seemingly competitive raw position error (e.g. 1.259 m median on the full dataset vs. 1.308 m for LangLoc w/o dialog). This is entirely an artefact of small indoor rooms: the floor centroid is mechanically close to every point in a compact 3–5 m space. The midpoint carries no practical value, as it conveys neither a meaningful within-room position nor any viewing direction, and its IoU is undefined.

The Top-10 metric exposes the difference: the correct position is almost always among LangLoc’s high-scoring cells (0.95 m median), well below the midpoint’s raw error. LangLoc commits to a specific floor region near the described objects, which can be wrong when several object clusters match the description, but the correct region is reliably represented in the posterior. With dialog, LangLoc surpasses the midpoint on every metric including raw position error.

7.2.2 Visibility Scoring as Implicit Grounding

The fine-localization stage performs an implicit form of spatial grounding: by scoring each grid cell according to how many described objects are simultaneously visible, the system identifies the region of the floor from which the described viewpoint is geometrically plausible. This is fundamentally different from object-level grounding tasks (e.g. ScanRefer (Chen et al., 2020)), where the goal is to localise a single referred object. Here, the observer’s position is inferred from the *joint* visibility of multiple objects, a collective constraint that narrows the feasible region more than any single object could.

The proximity bonus (5.31) further refines this by favouring positions close to the mentioned objects, reflecting the commonsense assumption that described objects are likely within conversational range rather than at the far end of a room.

7.2.3 Cross-Dataset Generalisation

The method generalises across datasets without retraining: the same pipeline produces comparable results on 3RScan (Top-10 median 0.94 m) and ScanNet (Top-10 median 1.07 m), despite differences in room layouts, annotation conventions, and mesh quality. This is because the fine-localization stage operates directly on the scene mesh and does not depend on learned parameters; it relies entirely on geometric ray-casting and Word2Vec label matching, both of which transfer across datasets.

7.3 Dialog-based Disambiguation

7.3.1 Impact of Dialog

The dialog module delivers the most dramatic improvements in the pipeline. On ScanNet, the median position error drops from 1.31 m to 0.07 m (a $\sim 19\times$ reduction) and the median angular error from 34.7° to 5.1° . On 3RScan, the gains are smaller but still substantial, with a 48% reduction in median position error and a near-doubling of IoU. The larger gains on ScanNet may reflect its richer annotation density, which provides more discriminative questions for the dialog module to exploit.

These improvements indicate that a small number of targeted yes/no questions is often sufficient to resolve the ambiguity that remains after a single free-form description. In particular, the information-gain question-selection strategy rapidly concentrates the posterior on a small set of plausible poses, so most successful episodes converge within only a few rounds. This shows that the dialog stage is not merely a minor refinement, but an effective mechanism for turning a broad semantic description into a precise pose estimate.

7.3.2 Limitations and Future Work

Several limitations remain and point directly to promising directions for future work. First, the pipeline depends on pre-built 3D models with semantic instance segmentation. Extending the approach to coarser or incrementally constructed representations, such as floor plans, schematic maps, or online scene graphs, would make it more practical in real deployments. Second, both the retrieval encoder and the fine-localization stage operate on closed label sets, so descriptions containing previously unseen object categories remain difficult to handle. Open-vocabulary extensions based on foundation models are therefore a natural next step.

Performance also degrades in large and cluttered scenes, where many viewpoints share similar object configurations and are difficult to distinguish using only object presence and simple pairwise relations. Addressing this limitation will likely require richer spatial reasoning and more adaptive question-selection policies. Moving beyond binary yes/no questioning may also be beneficial, but this would require a substantially richer posterior representation and more complex inference than the current dialog backend. In addition, the dialog stage still relies on a number of hand-tuned thresholds and hyperparameters, including parameters in the candidate-to-frame mapping, the likelihood model, and the question-selection heuristics. Reducing this dependence, or learning such parameters directly from data, would improve robustness and transfer across datasets.

Finally, the current pipeline is modular, with scene retrieval, fine localization, and dialog-based disambiguation designed and optimised as separate stages. Although this decomposition makes the system interpretable and easy to analyse, an end-to-end formulation that jointly optimises all stages could yield further gains. The present dialog evaluation is also only a partial proxy for real deployment: on 3RScan, answers are provided interactively by a human annotator, whereas on ScanNet they are generated by Qwen from ground-truth reference-frame metadata rather than from real user interaction. A broader human evaluation with noisy, uncertain, and potentially inconsistent responses would therefore be valuable for assessing how robust the dialog policy is under realistic conditions.

Conclusion

We have presented LangLoc, the first pipeline that goes beyond coarse scene retrieval to deliver fine-grained indoor localization from natural language alone. By combining a dual-branch GATv2 encoder for scene retrieval, visibility-based floor-grid scoring for pose estimation, and a Bayesian dialog module for disambiguation, LangLoc estimates a 2D position $\hat{c} \in \mathbb{R}^2$ and heading $\hat{\theta} \in \mathbb{S}^2$ from a free-form text description, without any image.

8.1 Summary of Contributions

Scene retrieval. The dual-branch GATv2 encoder with CLIP semantic features achieves 76.7% Top-1 recall on ScanScribe, surpassing the previous best (Text2SGM) by 8.1 percentage points. The CLIP-based encoding provides robustness to unseen text phrasing: under LLM-generated queries that reduce lexical overlap with database labels, LangLoc degrades by only 0.6 percentage points while Text2SGM collapses by 34 points. This robustness is critical for real-world deployment where user descriptions will inevitably differ from training-time phrasing.

Fine-grained localization. The visibility-based floor-grid scoring method achieves approximately 1 m median position error and meaningful heading estimates on the proposed LangLoc benchmark (13,000+ descriptions over 1,300+ scans). The 3D View IoU metric confirms that the method recovers both accurate position and orientation, achieving an order-of-magnitude improvement over the VLM baseline (0.147 vs. 0.018).

Dialog-based disambiguation. The interactive Bayesian dialog module delivers further dramatic gains. On ScanNet, a median position error of 0.07 m and angular error of 5.1° are achieved after just a few targeted yes/no questions, demonstrating that dialog can effectively resolve the spatial ambiguity inherent in a single free-form description.

Author Contributions

This project was developed collaboratively, with each author contributing equally to the overall development. Each component of the pipeline was led by a dedicated team member, as outlined below.

1. **Scene retrieval:** **Shirley Lau Feng Yi** developed the graph-based retrieval model, including the construction of text and scene graphs and the implementation of the GATv2 encoder for cross-modal matching.
2. **Dataset construction and codebase integration:** **Roham Zendehtdel Nobari** built the end-to-end pipeline that produced the *LangLoc* dataset, also contributed to the fine-localization module, wrote the integration code tying the three pipeline stages (Scene Retrieval, Fine Localization, Dialogue Disambiguity) into a single runnable system, and maintains the project repository.
3. **Fine localization:** **Shaurya Kishore Panwar** implemented the visibility-based localization module and its components, including text-to-scene object matching, candidate pose generation on the floor grid, ray-cast visibility scoring against the reconstructed mesh, FoV-consistent viewing direction estimation, and density-based pose aggregation.
4. **Dialog-based disambiguation:** **Abu Bakr Rahman Shaik** designed and implemented the dialog-based refinement module, including the Bayesian update mechanism and question selection strategy.

All authors contributed to system design, experimental evaluation, and writing of the report.

Hyperparameters and Architecture Details

This appendix collects the full hyperparameter tables and architecture specifications for all pipeline stages, supplementing the method descriptions in Chapter 5.

A.1 GATv2 Encoder Architecture

Table A.1 details the components of the dual-branch GATv2 encoder used for scene retrieval (Section 5.4.1). The total parameter count is approximately 2.3M.

Table A.1: GATv2 ENCODER ARCHITECTURE.

Component	Configuration
Node MLP (5.16)	Linear(518→256) + LayerNorm + GELU + Dropout(0.1) + Linear(256→256) + LayerNorm
Geometric branch (5.19)	3× GATv2Conv (256-dim, 2 heads, edge_dim=8) + residual skip connections + post-LayerNorm
Relation branch (5.20)	3× GATv2Conv (256-dim, 2 heads, edge_dim=512) + residual skip connections + post-LayerNorm
Gated fusion (5.21)	Sigmoid gate on $[g t]$ + $0.1 \times \text{MLP}([g t])$ residual + LayerNorm
Graph MLP (5.24)	Linear(768→256) + LayerNorm + GELU + Dropout(0.1) + Linear(256→256)

Node features (518-dim) concatenate: object centroid (3D), mean RGB colour (3D), and CLIP ViT-B/32 label embedding (512D). Text-derived graphs zero-pad the centroid and colour fields.

Geometric edge features (8-dim): $[\Delta c_{ij}, \|\Delta c_{ij}\|_2, r_i, r_j, 0, 0]$, where Δc_{ij} is the centroid offset and r_i, r_j are bounding radii.

A.2 Scene Retrieval Training

Table A.2 summarises the training hyperparameters for the scene retrieval encoder.

Table A.2: TRAINING HYPERPARAMETERS FOR SCENE RETRIEVAL. *Training hyperparameters for scene retrieval.*

Parameter	Pre-training	Fine-tuning
Optimizer	AdamW	AdamW
Learning rate	10^{-3}	10^{-5}
Weight decay	10^{-4}	10^{-4}
Epochs	100	30
Batch size	8	8
Temperature (InfoNCE)	0.1	0.1
Gradient clipping	max_norm = 1.0	max_norm = 1.0
LR schedule	10% warmup + cosine	CosineAnnealingLR
Negative ratio	50%	50%

A.3 Retrieval Scoring Weights

The composite retrieval score (5.25) uses the weights in Table A.3, tuned on a validation split. Label overlap uses base labels with spatial prefixes removed (e.g. “north chair” → “chair”).

Table A.3: RETRIEVAL SCORING WEIGHTS. *Retrieval scoring weights.*

Component	Symbol	Value
Graph embedding similarity	w_{emb}	0.33
Scene CLIP similarity	w_{glob}	0.33
Label overlap F1	w_{jac}	0.34

A.4 Fine-Localization Hyperparameters

Table A.4 summarises all method-facing hyperparameters used by the fine-localization pipeline. For parameters with multiple possible settings, the selected value is underlined.

A.5 Dialog Hyperparameters

A.5.1 Likelihood Model

Table A.5 lists the hyperparameters of the likelihood model shared across all three dialog backends (Section 5.6.3). The reliability parameter α_q controls how strongly the model trusts the underlying yes/no proposition for a question of type q . We use separate values for label and relation questions, with a lower value for relation questions because relation cues are noisier than binary object-presence cues. In the current system, these values are treated as fixed hyperparameters rather than being estimated from held-out calibration data.

The parameters p_u^{base} and p_u^{unans} control the probability of the answer *unknown*. The former specifies the baseline unknown mass even when a question is answerable, while the latter increases this mass when the question is not answerable from the current frame. We set p_u^{base} higher

Table A.4: FINE-LOCALIZATION HYPERPARAMETERS. *Fine-localization hyperparameters used during evaluation on the full 3RScan dataset.*

Parameter	Symbol	Value(s)
General		
Dataset	—	{ <u>3rscan</u> , <u>scannet</u> }
Query selection and grounding		
Frame selection policy	—	{first, index, random, max_visible, max_pixels, <u>all</u> }
Grounding threshold	γ	0.7
Scene-graph matching		
Top- K matched objects	K_{obj}	10
Dynamic Top- K	—	{disabled, <u>enabled</u> }
Match score threshold	s_{min}	0.1
Grid localization		
Grid spacing	Δ	0.25 m
Eye height	h	1.6 m
Softmax temperature	τ	1.5
Probability epsilon	ε	10^{-6}
Distance-bonus weight	λ_d	0.5
Distance-bonus decay	α_d	2.0 m
Prediction strategy	—	{argmax, <u>weighted</u> , random}
Kernel bandwidth	σ	1.0 m
Max cluster candidates	N_{cluster}	50
Orientation / FoV aggregation		
Horizontal FOV	hFoV	39.31° (Dataset Dependent)
Vertical FOV	vFoV	64.76° (Dataset Dependent)
Arrow stride	s_{arrow}	2
Top FoV poses retained	N_{pose}	5
Evaluation metrics		
Top- K min-distance	K_{min}	10

for relation questions than for label questions, based on the intuition that relation judgments are typically more ambiguous than binary object-presence judgments. The scales τ_{vis} and τ_{ans} convert label salience into soft truth and answerability probabilities, respectively.

Table A.5: LIKELIHOOD MODEL HYPERPARAMETERS. *Likelihood model hyperparameters.*

Parameter	Label questions	Relation questions
Reliability α_q	0.82	0.70
Unknown mass (answerable) p_u^{base}	0.05	0.15
Unknown mass (unanswerable) p_u^{unans}	0.90	0.90
Visibility scale τ_{vis}	0.20	0.20
Answerability scale τ_{ans}	0.10	0.10

A.5.2 Stopping Criteria

Table A.6 lists the stopping criteria for the dialog module.

The dialog terminates when the posterior concentrates above τ_{conf} on a single frame, or after

Table A.6: DIALOG STOPPING CRITERIA. *Dialog stopping criteria.*

Parameter	Value
Confidence threshold τ_{conf}	0.85
Minimum rounds	2
Maximum rounds	12
Auto-relax	Yes (progressively widens thresholds if question pool is exhausted)

the maximum number of rounds. At least 2 rounds are always completed regardless of posterior concentration. If the question pool is exhausted before reaching τ_{conf} , balance thresholds are progressively relaxed to admit previously filtered questions.

A.5.3 Question Selection Parameters

Common labels (floor, wall, ceiling, room, baseboard, carpet) are excluded entirely from the question pool. For remaining label questions, information-gain scores are multiplied by an IDF boost:

$$s'(q) = s(q) \cdot (1 + w_{\text{idf}} \cdot \text{IDF}(\ell)), \quad \text{IDF}(\ell) = \log \frac{F + 1}{\text{df}(\ell) + 1}, \quad (\text{A.1})$$

where $\text{df}(\ell)$ counts how many pooled frames contain label ℓ , F is the pool size, and $w_{\text{idf}} = 1.0$. Relation questions receive a bonus factor $(1 + r_{\text{bonus}})$ with $r_{\text{bonus}} = 0.25$. Questions with $P(\text{yes}) \notin [0.01, 0.99]$ under the current posterior are excluded.

A.6 Qwen-based Answering Protocol

For quantitative evaluation on ScanNet, dialog answers are generated automatically rather than by a human annotator. Concretely, we use a locally hosted large language model, Qwen2.5-1.5B-Instruct, to answer each dialog question from ground-truth reference-frame metadata. The model is run on local compute rather than accessed through an external API.

For each evaluated scene, we first identify the reference frame associated with the example and load its frame-level metadata. From this frame, we construct a compact textual context containing three elements: the frame description, the list of visible object labels, and the spatial relations stored for that frame. Each dialog question is then converted into a natural-language prompt and passed to Qwen together with this frame context. The model is instructed to answer with exactly one of “yes”, “no”, or “uncertain”, based only on the provided frame description and metadata. The generated response is finally mapped to the dialog symbols $\bar{y}$, n , and u .

Dialog Backend Architectures

The main text (Section 5.6) describes the dialog module at a conceptual level, presenting the Bayesian update rule and question selection strategy. In practice, we implement three backend architectures that share the same likelihood model and question selection logic but differ in their state-space representation. This appendix details all three and justifies the selection of Backend A3 as the default.

B.1 Backend A1: Candidate Posterior

Backend A1 maintains a discrete posterior $p(\mathbf{q})$ directly over the N candidate poses from the fine-localization grid. Each candidate $\mathbf{q}_i = (\mathbf{c}_i, \theta_i)$ is linked to its $K=15$ nearest reference frames via a mapping matrix $W \in \mathbb{R}^{N \times F}$, where:

$$W_{ij} = \frac{\exp(-d_{ij}^2/2\sigma^2)}{\sum_{j'} \exp(-d_{ij'}^2/2\sigma^2)}, \quad \sigma = 0.25 \text{ m}. \quad (\text{B.1})$$

To evaluate a question for candidate $\mathbf{q}_i$, semantic attributes are aggregated from the mapped frames: $p_{\text{vis}}(\mathbf{q}_i) = \sum_j W_{ij} \cdot p_{\text{vis}}(f_j)$. The frame posterior is derived as $p_f = W^\top p$.

Strengths. Explicitly models the candidate-to-frame relationship; the posterior lives in the same space as the localization grid.

Weaknesses. Requires maintaining and multiplying through the $N \times F$ mapping matrix at each update; semantic signals are smoothed through the aggregation, potentially losing discriminative power.

B.2 Backend A2: Particle Filter

Backend A2 represents the posterior with $P=256$ weighted particles in continuous 3D space. Particles are initialised by sampling candidates proportionally to the prior, with Gaussian jitter ($\sigma_{\text{jitter}} = 0.07 \text{ m}$).

At each update:

1. For each particle, find $K=10$ nearest reference frames and compute Gaussian kernel weights ($\sigma = 0.25 \text{ m}$).

2. Aggregate frame-level semantics via these weights to obtain per-particle truth and answerability probabilities.
3. Update particle weights via Bayes' rule.
4. Resample particles with replacement according to updated weights, then apply fresh jitter.

Strengths. Operates in continuous space; allows smooth spatial interpolation; resampling prevents weight collapse on wrong modes.

Weaknesses. KNN lookup at every update adds computational overhead; resampling jitter introduces noise; particle count limits resolution.

B.3 Backend A3: Frame Posterior (Default)

Backend A3 maintains the posterior directly over F_{pool} pooled reference frames. The initial prior and Bayesian update follow the same formulation as in Section ?? and (??), but crucially, frame-level semantics (label salience dictionaries, relation sets) are accessed *directly*: no aggregation or interpolation is needed, and the likelihood $\mathcal{L}(a | j)$ uses the exact visibility and relation metadata stored per frame.

Why A3 is the default. We select A3 as the default backend for three reasons:

1. **Semantic fidelity.** A3 accesses ground-truth frame semantics directly, whereas A1 and A2 must interpolate or aggregate through a mapping layer, which can smooth away discriminative signals.
2. **Simplicity and speed.** Each Bayesian update requires only a direct dictionary lookup per frame, with no matrix multiplications (A1) or KNN queries (A2).
3. **Empirical performance.** In our shared evaluation mode, where all three backends receive the same questions, A3 consistently achieves the highest posterior concentration and lowest final position error, as shown in Tables B.1 and B.2.

B.4 Backend Comparison Results

Tables B.1 and B.2 compare all three backends on the 100-scene subsets. The baseline row shows fine-localization performance without any dialog (i.e. the initial posterior before the first question). All three backends receive the same oracle-generated questions and answers, so differences in performance are attributable solely to the posterior representation.

Table B.1: BACKEND COMPARISON ON 3RSCAN. *Backend comparison on 3RScan (100-scene subset). Best in **bold**, second best underlined.*

Method	Top-10 Pos. (m) ↓		Pos. (m) ↓		Angle (deg) ↓		3D IoU ↑
	Mean	Med.	Mean	Med.	Mean	Med.	Mean
Baseline (no dialog)	1.037	0.941	1.712	1.551	<u>46.07</u>	<u>37.24</u>	<u>0.172</u>
Candidate Posterior (A1)	1.537	1.203	1.716	1.292	74.32	79.04	0.109
Particle Filter (A2)	<u>1.387</u>	<u>1.189</u>	<u>1.581</u>	<u>1.255</u>	57.76	51.52	0.126
Frame Posterior (A3)	0.983	0.890	0.926	0.799	39.52	33.37	0.342

Table B.2: BACKEND COMPARISON ON SCANNET. *Backend comparison on ScanNet (100-scene subset). Best in **bold**, second best underlined.*

Method	Top-10 Pos. (m) ↓		Pos. (m) ↓		Angle (deg) ↓		3D IoU ↑
	Mean	Med.	Mean	Med.	Mean	Med.	Mean
Baseline (no dialog)	<u>1.254</u>	<u>1.065</u>	1.676	<u>1.314</u>	42.67	34.66	<u>0.236</u>
Candidate Posterior (A1)	1.527	1.270	1.764	1.599	43.05	34.86	0.187
Particle Filter (A2)	1.378	1.147	<u>1.651</u>	1.436	<u>34.51</u>	<u>22.22</u>	0.203
Frame Posterior (A3)	0.532	0.211	0.593	0.069	23.77	5.12	0.593

Analysis. A3 dominates all metrics on both datasets. On 3RScan, A3 reduces the median position error from 1.551 m (baseline) to 0.799 m and nearly doubles the 3D IoU from 0.172 to 0.342. A1 and A2 both fail to improve over the baseline in position error on 3RScan and actually degrade angular accuracy, with A1 producing near-random headings (79° median). This suggests that the candidate-to-frame aggregation in A1 smooths away the discriminative signal needed for effective posterior updates.

On ScanNet, the performance gap is even more pronounced: A3 achieves a median position error of just 0.069 m and an IoU of 0.593, while A2 reaches only 1.436 m and 0.203. A2 outperforms A1 on angular metrics (median 22.2° vs. 34.9°), indicating that particle-based continuous interpolation preserves directional information better than the discrete candidate mapping. However, neither A1 nor A2 approaches A3’s accuracy.

The consistent advantage of A3 stems from its direct access to frame-level semantics: each Bayesian update uses the exact visibility and relation metadata stored per frame, without any interpolation or aggregation that could smooth away discriminative signals. This makes A3 both the simplest and the most effective backend.

List of Figures

1.1	LangLoc overview	2
2.1	Graph attention mechanism	6
2.2	Static vs. dynamic attention	8
2.3	CLIP contrastive pre-training	10
4.1	Per-scene outputs of the LangLoc data pipeline	22
5.1	Coarse Retrieval Pipeline Overview	30
5.2	Fine-localization pipeline	35
5.3	Direction field	37
5.4	Dialog-based disambiguation overview	40
6.1	Qualitative frustum overlap on 3RScan	51
6.2	Qualitative frustum overlap on ScanNet	51
6.3	Dialog trace for scene0002_00	52

List of Tables

6.1	Scene retrieval Recall@ k on ScanScribe (10-scene pool)	47
6.2	Scene retrieval Recall@ k on the full 55-scene test set	48
6.3	Scene retrieval Recall@ k with LLM-generated queries	48
6.4	Fine localization on 3RScan	50
6.5	Fine localization on ScanNet	50
6.6	Fine localization on the full LangLoc dataset	50
6.7	Example dialog transcript for scene0002_00	51
A.1	GATv2 encoder architecture	61
A.2	Training hyperparameters for scene retrieval	62
A.3	Retrieval scoring weights	62
A.4	Fine-localization hyperparameters	63
A.5	Likelihood model hyperparameters	63
A.6	Dialog stopping criteria	64
B.1	Backend comparison on 3RScan	67
B.2	Backend comparison on ScanNet	67

Abbreviations

3DSSG	3D Semantic Scene Graphs
CLIP	Contrastive Language–Image Pre-training
CNN	Convolutional Neural Network
DPP	Determinantal Point Process
FoV	Field of View
GAT	Graph Attention Network
GATv2	Graph Attention Network, version 2
GCN	Graph Convolutional Network
GELU	Gaussian Error Linear Unit
GNN	Graph Neural Network
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
IDF	Inverse Document Frequency
InfoNCE	Information Noise-Contrastive Estimation
IoU	Intersection over Union
KNN	K -Nearest Neighbours
LLM	Large Language Model
MAP	Maximum A Posteriori
MLP	Multi-Layer Perceptron
NP-hard	Non-deterministic Polynomial-time hard
PCA	Principal Component Analysis
PnP	Perspective-n-Point
RANSAC	RANdom SAmple Consensus
RGB-D	Red, Green, Blue and Depth
ViT	Vision Transformer
VLM	Vision–Language Model
Word2Vec	Word-to-Vector embedding

Bibliography

- Achlioptas, P., Abdelreheem, A., Xia, F., Elhoseiny, M., and Guibas, L. J. (2020). ReferIt3D: Neural listeners for fine-grained 3d object identification in real-world scenes. In *16th European Conference on Computer Vision (ECCV)*.
- Agnolucci, L., Galteri, L., and Bertini, M. (2025). Quality-aware image-text alignment for opinion-unaware image quality assessment.
- Ali-bey, A., Chaib-draa, B., and Giguère, P. (2023). MixVPR: Feature mixing for visual place recognition. pages 14254–14263.
- Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and van den Hengel, A. (2018). Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. pages 3674–3683.
- Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. pages 5297–5307.
- Armeni, I., He, Z.-Y., Gwak, J., Zamir, A. R., Fischer, M., Malik, J., and Savarese, S. (2019). 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5664–5673.
- Azuma, D., Miyanishi, T., Kurita, S., and Kawanabe, M. (2022). Scanqa: 3d question answering for spatial scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Berton, G., Masone, C., and Caputo, B. (2022). Rethinking visual geo-localization for large-scale applications. pages 4878–4888.
- Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B. (2010). *Polygon mesh processing*. CRC press.
- Brachmann, E., Cavallari, T., and Prisacariu, V. A. (2023). Accelerated coordinate encoding: Learning to relocalize in minutes using RGB and poses. pages 5044–5053.
- Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., and Rother, C. (2017). DSAC – differentiable RANSAC for camera localization. pages 6684–6692.
- Brachmann, E. and Rother, C. (2018). Learning less is more – 6D camera localization via 3D surface regression. pages 4654–4662.
- Brahmbhatt, S., Gu, J., Kim, K., Hays, J., and Kautz, J. (2018). Geometry-aware learning of maps for camera localization. pages 2616–2625.

- Brody, S., Alon, U., and Yahav, E. (2022). How attentive are graph attention networks? In *International Conference on Learning Representations*.
- Chen, D. Z., Chang, A. X., and Nießner, M. (2020). Scanrefer: 3d object localization in rgb-d scans using natural language. *16th European Conference on Computer Vision (ECCV)*.
- Chen, J., Barath, D., Armeni, I., Pollefeys, M., and Blum, H. (2024). “where am i?” scene retrieval with language. In *European Conference on Computer Vision*, pages 201–220. Springer.
- Chen, X., Li, L.-J., Fei-Fei, L., and Gupta, A. (2018). Iterative visual reasoning beyond convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7239–7248.
- Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., and Nießner, M. (2017a). Scan-net: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*.
- Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., and Theobalt, C. (2017b). Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4):1.
- DeTone, D., Malisiewicz, T., and Rabinovich, A. (2018). SuperPoint: Self-supervised interest point detection and description. pages 224–236.
- Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272.
- Glassner, A. S. (1989). *An introduction to ray tracing*. Morgan Kaufmann.
- Gu, J., Hu, H., Wang, L., Wei, Y., and Dai, J. (2018). Learning region features for object detection. In *Proceedings of the european conference on computer vision (ECCV)*, pages 381–395.
- Gu, Q., Kuwajerwala, A., Morin, S., Jatavallabhula, K. M., Sen, B., Agarwal, A., Rivera, C., Paul, W., Ellis, K., Chellappa, R., Gan, C., de Melo, C., Tenenbaum, J. B., Torralba, A., Shkurti, F., and Paull, L. (2024). ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning. In *IEEE Int. Conf. Robot. Automat. (ICRA)*.
- Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.
- Hausler, S., Garg, S., Xu, M., Milford, M., and Fischer, T. (2021). Patch-NetVLAD: Multi-scale fusion of locally-global descriptors for place recognition. pages 14141–14152.
- Hong, Y., Wu, Q., Qi, Y., Rodriguez-Opazo, C., and Gould, S. (2021). VLN-BERT: A recurrent vision-and-language BERT for navigation. pages 1643–1653.
- Hong, Y., Zhen, H., Chen, P., Zheng, S., Du, Y., Chen, Z., and Gan, C. (2023). 3D-LLM: Injecting the 3D world into large language models.
- Hu, H., Gu, J., Zhang, Z., Dai, J., and Wei, Y. (2018). Relation networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3588–3597.
- Kendall, A., Grimes, M., and Cipolla, R. (2015). PoseNet: A convolutional network for real-time 6-DOF camera relocalization. pages 2938–2946.
- Kerr, J., Kim, C. M., Goldberg, K., Kanazawa, A., and Tancik, M. (2023). LERF: Language embedded radiance fields. pages 19729–19739.

- Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.
- Kolmet, M., Zhou, Q., Osep, A., and Leal-Taixé, L. (2022). Text2Pos: Text-to-point-cloud cross-modal localization. pages 15172–15181.
- Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286.
- Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations*.
- Ma, X., Yong, S., Zheng, Z., Li, Q., Liang, Y., Zhu, S.-C., and Huang, S. (2023). SQA3D: Situated question answering in 3D scenes.
- Miao, Y., Engelmann, F., Vysotska, O., Tombari, F., Pollefeys, M., and Baráth, D. B. (2024). Scene-graphloc: Cross-modal coarse visual localization on 3d scene graphs. In *European Conference on Computer Vision*, pages 127–150. Springer.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: A versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163.
- Narasimhan, M., Lazebnik, S., and Schwing, A. (2018). Out of the box: Reasoning with graph convolution nets for factual visual question answering. *Advances in neural information processing systems*, 31.
- Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee.
- Norcliffe-Brown, W., Vafeias, S., and Parisot, S. (2018). Learning conditioned graph structures for interpretable visual question answering. *Advances in neural information processing systems*, 31.
- Peng, S., Genova, K., Jiang, C. M., Tagliasacchi, A., Pollefeys, M., and Funkhouser, T. (2023). OpenScene: 3D scene understanding with open vocabularies. pages 815–824.
- Qi, Y., Wu, Q., Anderson, P., Wang, X., Wang, W. Y., Shen, C., and van den Hengel, A. (2020). REVERIE: Remote embodied visual referring expression in real indoor environments. pages 9982–9991.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021a). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021b). Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020 [cs].
- Sarkar, S. D., Miksik, O., Pollefeys, M., Barath, D., and Armeni, I. (2023). Sgaligner: 3d scene alignment with scene graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21927–21937.

- Sarlin, P.-E., Cadena, C., Siegwart, R., and Dymczyk, M. (2019). From coarse to fine: Robust hierarchical localization at large scale. pages 12716–12725.
- Sarlin, P.-E., DeTone, D., Malisiewicz, T., and Rabinovich, A. (2020). SuperGlue: Learning feature matching with graph neural networks. pages 4938–4947.
- Sarlin, P.-E., DeTone, D., Yang, T.-Y., Avetisyan, A., Straub, J., Malisiewicz, T., Rota Buló, S., Newcombe, R., Kotschieder, P., and Balntas, V. (2023). OrienterNet: Visual localization in 2D public maps with neural matching. pages 21632–21642.
- Sarlin, P.-E., Unagar, A., Lärsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Kahl, L., and Sattler, T. (2021). Back to the feature: Learning robust camera localization from pixels to pose. pages 3247–3257.
- Sattler, T., Leibe, B., and Kobbelt, L. (2017). Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(9):1744–1756.
- Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., and Torii, A. (2018). InLoc: Indoor visual localization with dense matching and view synthesis. pages 7199–7208.
- Teney, D., Liu, L., and van den Hengel, A. (2017). Graph-structured representations for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Thomason, J., Murray, M., Cakmak, M., and Zettlemoyer, L. (2020). Vision-and-dialog navigation. In Kaelbling, L. P., Kragic, D., and Sugiura, K., editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 394–406. PMLR.
- van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In *International Conference on Learning Representations*.
- Wald, I., Boullos, S., and Shirley, P. (2007). Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Transactions on Graphics (TOG)*, 26(1):6–es.
- Wald, J., Avetisyan, A., Navab, N., Tombari, F., and Nießner, M. (2019). Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings IEEE International Conference on Computer Vision (ICCV)*.
- Wald, J., Dharm, H., Navab, N., and Tombari, F. (2020). Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., et al. (2024). Qwen2-VL: Enhancing vision-language model’s perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.
- Wang, P., Wu, Q., Shen, C., Dick, A., and van den Hengel, A. (2018). Fvqa: Fact-based visual question answering requiring multi-hop reasoning. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 40, pages 2413–2427.
- Wu, S.-C., Wald, J., Tateno, K., Navab, N., and Tombari, F. (2021). SceneGraphFusion: Incremental 3D scene graph prediction from RGB-D sequences. pages 7515–7525.

- Zhang, C., Li, M., Budvytis, I., and Liwicki, S. (2024). Dialog: An iterative approach to embodied dialog localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12585–12593.
- Zhou, G., Hong, Y., and Wu, Q. (2024). NavGPT: Explicit reasoning in vision-and-language navigation with large language models. pages 7641–7649.
- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.
- Zhou, Q.-Y., Park, J., and Koltun, V. (2018). Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*.
- Zhu, Z., Ma, X., Chen, Y., Deng, Z., Huang, S., and Li, Q. (2023). 3d-vista: Pre-trained transformer for 3d vision and text alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2911–2921.